PB90-172560

MOUSE - A COMPUTERIZED UNCERTAINTY ANALYSIS SYSTEM

(U.S.) Environmental Protection Agency
Cincinnati, OH

Jan 90

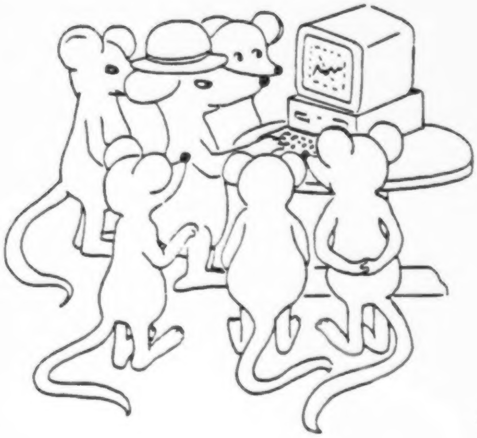
U.S. Department of Commerce
National Technical Information Service
NTIS

PB90-172560

EPA/600/8-89/102a
January 1990

MOUSE
A COMPUTERIZED UNCERTAINTY ANALYSIS SYSTEM
OPERATIONAL MANUAL

by
Albert J. Klee
Risk Reduction Engineering Laboratory
U.S. Environmental Protection Agency
Cincinnati, Ohio 45268



RISK REDUCTION ENGINEERING LABORATORY
OFFICE OF RESEARCH AND DEVELOPMENT
U.S. ENVIRONMENTAL PROTECTION AGENCY
CINCINNATI, OHIO 45268

REPRODUCED BY
U.S. DEPARTMENT OF COMMERCE
NATIONAL TECHNICAL INFORMATION SERVICE
SPRINGFIELD, VA 22161

TECHNICAL REPORT DATA

(Please read Instructions on the reverse before completing)

1. REPORT NO. EPA/600/8-89/102a	2.	3. RECIPIENT'S ACCESSION NO. PB90-172560
4. TITLE AND SUBTITLE MOUSE - A Computerized Uncertainty Analysis System		5. REPORT DATE January 1990
		6. PERFORMING ORGANIZATION CODE
7. AUTHOR(S) Albert J. Klee		8. PERFORMING ORGANIZATION REPORT NO. EPA/SW/DK-90/019a
9. PERFORMING ORGANIZATION NAME AND ADDRESS Risk Reduction Engineering Laboratory--Cincinnati OH Office of Research and Development U.S. Environmental Protection Agency Cincinnati, OH		10. PROGRAM ELEMENT NO. D109N3202
		11. CONTRACT/GRANT NO. InHouse
12. SPONSORING AGENCY NAME AND ADDRESS Risk Reduction Engineering Laboratory--Cincinnati, OH Office of Research and Development U.S. Environmental Protection Agency Cincinnati, OH		13. TYPE OF REPORT AND PERIOD COVERED Complete
		14. SPONSORING AGENCY CODE EPA/600/14

15. SUPPLEMENTARY NOTES

P.O. Albert J. Klee Phone: (513) 569-7493 FTS 684-7493

For diskette see: PB90-501370

16. ABSTRACT

Environmental engineering calculations involving uncertainties are far beyond the capabilities of hand analysis for any but the simplest of models. There exist a number of computer simulation languages that certainly can do the job but learning such languages and implementing them quickly and error-free on personal computers is not the quickest or easiest of tasks. MOUSE (an acronym for Modular Oriented Uncertainty SystEm) deals with the problem of uncertainties in models that consist of one or more algebraic equations. It was especially designed for use by those with little or no knowledge of computer languages or programming. It is compact (and thus can be run on almost any personal computer), easy and fast to learn, and has most of the features for substantive uncertainty analysis (built-in probability distributions, plotting and graphing capabilities, sensitivity analysis, interest functions for cost analyses, etc.). Moreover, a series of unique companion utility programs not only write much of the necessary computer code for the user and help in analyzing sample data to determine the probability distributions that best suit that data, but they check each program for errors in syntax, and assist in finding logical errors in the models under study.

17. KEY WORDS AND DOCUMENT ANALYSIS

a. DESCRIPTORS	b. IDENTIFIERS/OPEN ENDED TERMS	c. COSATI Field/Group

18. DISTRIBUTION STATEMENT RELEASE TO PUBLIC	19. SECURITY CLASS *(This Report)* UNCLASSIFIED	21. NO. OF PAGES 233
	20. SECURITY CLASS *(This page)* UNCLASSIFIED	22. PRICE A11

DISCLAIMER

This report has been reviewed by the U.S. Environmental Protection Agency and approved for publication. Approval does not signify that the contents necessarily reflect the views and policies of the U.S. Environmental Protection Agency, nor does mention of trade names or commercial products constitute endorsement or recommendation of use.

FOREWORD

Today's rapidly developing and changing technologies and industrial products and practices frequently carry with them the increased generation of materials that, if improperly dealt with, can threaten both public health and the environment. The U.S. Environmental Protection Agency is charged by Congress with protecting the Nation's land, air, and water resources. Under a mandate of national environmental laws, the agency strives to formulate and implement actions leading to a compatible balance between human activities and the ability of natural systems to support and nurture life. These laws direct the EPA to perform research to define our environmental problems, measure the impacts, and search for solutions.

The Risk Reduction Engineering Laboratory is responsible for planning, implementing, and managing research, development, and demonstration programs to provide an authoritative, defensible engineering basis in support of the policies, programs, and regulations of the EPA with respect to drinking water, wastewater, pesticides, toxic substances, solid and hazardous wastes, and Superfund-related activities. This publication is one of the products of that research and provides a vital communication link between the researcher and the user community.

This manual describes a system, called MOUSE (for Modular Oriented Uncertainty SystEm), for dealing with the computational problems of uncertainty, specifically in models that consist of a set of one or more algebraic equations. Since such models are frequently encountered in the fields of environmental science, risk analysis, economics, and engineering, the system has broad application throughout these fields. No prior programming knowledge is needed to learn MOUSE and, because of its transportability and compactness, the system can be run on a wide variety of personal computers available to the U.S. Environmental Protection Agency and/or its contractors and grantees.

For further information, please contact the Waste Minimization, Destruction and Disposal Research Division of the Risk Reduction Engineering Laboratory.

E. Timothy Oppelt, Director
Risk Reduction Engineering Laboratory

ABSTRACT

Environmental engineering calculations involving uncertainties, either in the model itself or in the data, are far beyond the capabilities of conventional analysis for any but the simplest of models. There exist a number of general-purpose computer simulation languages, using Monte Carlo methods, that are capable of such analysis, but these languages are difficult to learn and to implement quickly.

MOUSE (an acronym for Modular Oriented Uncertainty SystEm) deals with the problem of uncertainties in models that consist of one or more algebraic equations. It was especially designed for use by those with little or no knowledge of computer languages, programming, or simulation. It is designed to be run on almost any personal computer, easy and fast to learn, and has all of the features needed for substantive uncertainty analysis (built-in probability distributions, plotting and graphing capabilities, sensitivity analysis, interest functions for cost analyses, etc.). Moreover, a series of unique companion utility programs write much of the necessary computer code for the user, help in analyzing sample data to determine the probability distributions that best fit those data, check each program for errors in syntax, and assist in finding logical errors in the model that is subject to uncertainty.

Some typical examples of the use of MOUSE within the U.S. Environmental Protection Agency include: studying the migration of pollution plumes in streams, establishing regulations for hazardous wastes in landfills, and estimating pollution control costs.

PREFACE

This manual describes, using a new personal computer-based programming language called MOUSE (for Modular Oriented Uncertainty SystEm), how to deal with the problem of uncertainty, specifically in models that consist of a set of one or more algebraic equations. Such models are frequently encountered in the fields of environmental science, risk analysis, economics, and engineering. No prior programming knowledge is needed to learn MOUSE and, because of its transportability and compactness, the system can be run on a wide variety of PC-DOS or MS-DOS personal computers available to the U.S. Environmental Protection Agency and/or its contractors and grantees. (The basic requirement is that the system have a FORTRAN compiler.)

The manual introduces the concept of uncertainty and its effects, discuss the implications of traditional solutions to the problem, and compares the alternative solutions available. A step-by-step approach, via examples, is taken to introduce the various features of MOUSE. The reader's understanding of these features is tested by numerous exercises, to which the solutions are given. Writing computer programs is often a tedious process, and one that is seldom error-free. To this end, several utilities are provided that make life easier for the user. One utility, TRAP, actually writes a great deal of the program while another, CHECKER, examines it for errors. Two additional utilities, TRACER and MSHELP, provide additional assistance in writing and debugging MOUSE programs.

The determination of probability distributions is perhaps the most difficult concept for newcomers to the field to master. The manual explores the methods by which the user can determine the probability distributions to be used in his model, whether they be subjective or based upon empirical data. In addition, an additional computer-based system, called IMP (for Interactive Modeler for Probabilities), is introduced as an aid in performing the data analysis required in the process.

The concept of simulation is not a new one, of course, but any attempt to apply simulation usually entails utilizing professionals who are familiar with simulation and simulation languages or else training in-house or contractor staff to do the same. Limited resources often preclude the former; the latter is not cost-effective unless a professional is expected to devote a major portion of his or her time to such matters. The concepts of simulation, coupled with the necessity to learn one of the standard simulation languages such as SIMSCRIPT or SLAM, are not skills transferable within a short period of time, however. Furthermore, without constant reinforcement through frequent use, such skills are quickly forgotten. MOUSE, on the other hand, was designed to make the writing of computer simulation models, based upon algebraic equations, as simple as possible.

Although some previous exposure to probability and statistics would be helpful, readers without formal training in these areas will not find the material difficult. The exposition has deliberately been kept on a tutorial level to facilitate implementation.

NOTE: The author would appreciate it if any errors found in this manual and/or bugs in any of the programs are reported to:

Albert J. Klee
Risk Reduction Engineering Laboratory
US Environmental Protection Agency
26 Martin Luther King Drive
Cincinnati, Ohio 45268

513/569-7493

CONTENTS

Page

Chapter 1 EXPLORING UNCERTAINTY

1.1 Defining the Problem.............................. 1
1.2 Consequences of Failure to Deal With Uncertainty.. 1
1.3 Traditional Approaches to Uncertainty............. 2
1.4 Problems with Traditional Approaches.............. 3
1.5 Alternative Solutions to the Uncertainty Problem.. 5
1.6 Traditional Sensitivity and Uncertainty Sensitivity Analysis Compared.................. 9
1.7 Languages For A Computerized Uncertainty Analysis System.................................. 9
Addendum.. 15
References.. 17

Chapter 2 EXAMPLES OF UNCERTAINTY ANALYSIS

2.1 A Cost Example.................................... 18
2.2 The Input Variables............................... 19
2.3 The "Best Value" Approach......................... 21
2.4 The MOUSE Oil Reclamation Program And Results.................................... 21
2.5 A Regulatory Example.............................. 23
2.6 The MOUSE Regulatory Program and Results.......... 28

Chapter 3 THE MOUSE LANGUAGE

3.1 Overview of MOUSE Programs........................ 31
3.2 Arithmetic.. 35
3.3 Restricted Names.................................. 38
3.4 Stochastic Input Variables........................ 39
Exercises... 39
Reference... 39

Chapter 4 MOUSE LIBRARY PROBABILITY FUNCTIONS

4.1 Introduction...................................... 40
4.2 Empirical Probability Functions................... 40
4.3 A Semi-Empirical Probability Function: The Step Rectangular Function................... 43
4.4 Classical Probability Functions: Continuous....... 45
4.5 Classical Probability Functions: Discrete......... 49
4.6 Truncated Distributions........................... 51
4.7 Composite Distributions........................... 52
4.8 Examples of Probability Function Calls............ 54

4.9 Notes On The Generation Of Stochastic Variates in MOUSE and Checks on MOUSE Probability Call Syntax.. 55
4.10 Fitting Classical Probability Distributions With IMP.. 55
4.11 Fitting Empirical Continuous Probability Distributions With IMP........................... 63
Exercises... 67
References.. 69

Chapter 5 MOUSE LIBRARY INTEREST FUNCTIONS

5.1 Introduction.. 71
5.2 Compound Interest................................... 71
5.3 Gradient Series..................................... 74
5.4 Nominal Interest Rates.............................. 74
5.5 Continuous Compounding.............................. 75
5.6 Nominal Interest Rates, Continuous Compounding.... 76
5.7 Example of Probability Function Calls and MOUSE Interest Call Syntax............................ 77
Exercises... 78
Reference... 78

Chapter 6 THE MOUSE TABLE

6.1 Introduction.. 79
6.2 The MOUSE-Dimensioned Table......................... 80
6.3 The User-Dimensioned Table 81
6.4 The Short Table..................................... 82
6.5 The Dummy Table..................................... 82
6.6 User Table Scaling.................................. 83
6.7 Literal Names in MOUSE.............................. 83
6.8 Mixing Table Types.................................. 84
Exercises... 84
References.. 84

Chapter 7 USER CONTROL DECISIONS

7.1 Introduction.. 85
7.2 The Iteration Number Line........................... 85
7.3 The Snap Interval Line.............................. 85
7.4 The Seed Reset Line................................. 86
7.5 The Output Destination and Column Output Lines.... 88
7.6 Summary of the Specification Block................ 89
7.7 Multiple Runs: Matrices and Vectors............... 89
7.8 Multiple Runs: The MOUSE Pass..................... 92

7.9 Program Output Titles: The Name Line 93
7.10 The Program Listing Line........................ 95
7.11 Preparing and Checking Specification Block Lines.. 95
Exercises.. 95
Reference.. 96

Chapter 8 ADDITIONAL MOUSE FUNCTIONS

8.1 Introduction...................................... 97
8.2 The Plot Output Line.............................. 97
8.3 MOUSE Automated Sensitivity: Introduction......... 101
8.4 The Sensitivity Control Line...................... 102
8.5 Sensitivity Level Line............................ 104
Exercises.. 105

Chapter 9 CORRELATED STOCHASTIC INPUT VARIABLES

9.1 Introduction...................................... 106
9.2 Linear Correlation................................ 106
9.3 Bivariate Normal Distribution..................... 108
9.4 Bivariate Lognormal Distribution.................. 112
9.5 Using IMP to Analyze Bivariate Normal And Bivariate Lognormal Distributions............... 114
9.6 Time Series and Autocorrelation................... 116
Exercises.. 119
References... 119

Chapter 10 NON-UNIFORM CASH FLOWS, VECTORS, AND THE INTERNAL-RATE-OF-RETURN FUNCTION

10.1 Introduction..................................... 121
10.2 The Present Worth Sum Function.................. 123
10.3 Calculating The Elements of a Vector............. 124
10.4 The Use of the IF Statement in Vector Loops...... 125
10.5 Vector Loops and Order of Statements............. 126
10.6 The Vector Table................................. 126
10.7 Vectors, The MOUSE Pass, And Plots............... 131
Exercises.. 135

Chapter 11 Troubleshooting, Error Messages, and Help

11.1 Introduction..................................... 137
11.2 CHECKER, A Utility for Catching MOUSE Syntactical Errors............................. 137

11.3 A Case History.................................... 141
11.4. MOUSE Run-Time Error Checks...................... 143
11.5 The MOUSE Trace................................... 147
11.6 TRAP, A Utility For Preparing MOUSE Source Programs............................. 151
11.7 Using TRAP To Prepare MOUSE Data Files........... 156
11.8 MSHELP - Another MOUSE Help Utility.............. 158
Exercises.. 159

Chapter 12 FITTING PROBABILITY DISTRIBUTIONS TO SAMPLE DATA

12.1 Introduction..................................... 160
12.2 Goodness-of-fit Tests............................ 162
12.3 Subjective Determination Of Probability Distributions.................................... 163
12.4 The Variable Interval Method..................... 163
12.5 The Fixed Interval Method........................ 165
12.6 Other Methods.................................... 166
12.7 Improving Elicitation Methods.................... 168
12.8 Some Useful Distributions for Subjective Methods.. 172
References... 174

Appendix A: RUNNING MOUSE, IMP, AND TRAP PROGRAMS

A.1 Introduction...................................... 176
A.2 Running CHECKER, IMP, TRACER, and TRAP On PC-DOS or MS-DOS Computers..................... 177
A.3 Running MOUSE On PC-DOS or MS-DOS Computers....... 177

Appendix B: INSTALLATION AND CUSTOMIZATION NOTES FOR MOUSE

B.1 Installation...................................... 180
B.2 Compilation and Linking With Libraries............ 181
B.3 Adding Modules to MOUSE........................... 183
References... 184

Appendix C: SUMMARY TABLES

Table C.1: MOUSE Control Lines......................... 185
Table C.2: Alphabetical List of MOUSE Interest Functions............................... 186
Table C.3: Alphabetical List of MOUSE Probability Functions............................... 187
Table C.4: MOUSE Line Types............................ 188

Appendix D:

MOUSE Error Messages.................................. 189

Appendix E: ANSWERS TO EXERCISES

Chapter 3.. 192
Chapter 4.. 192
Chapter 5.. 193
Chapter 6.. 194
Chapter 7.. 194
Chapter 8.. 195
Chapter 9.. 195
Chapter 10....................................... 195
Chapter 11....................................... 198

Appendix F: MOUSE RANDOM NUMBER TESTER

F.1 Introduction.................................. 199
F.2 Gap Test...................................... 200
F.3 Two-dimensional serial test................... 201
F.4 Kolmogorov-smirnov test....................... 201
F.5 Poker test.................................... 201
F.6 Permutation test.............................. 201
F.7 Runs test..................................... 202
F.8 Maximum-of-T test............................. 202
F.9 D-square test................................. 202
F.10 Serial correlations and correlograms......... 202
F.11 Lag test..................................... 203

Index.. 204

LIST OF FIGURES

Number **Page**

1.1: Examples of Uncertainty. ....
1.2: Direct (Complete) Enumeration.......................... 4
1.3: Probability Calculus Approach.......................... 6
1.4: Monte Carlo Simulation................................. 8
1.5: Sensitivity Analysis: Traditional Method............... 10
1.6: Sensitivity Analysis: Uncertainty Method............... 10
2.1: MOUSE Program for the Waste Oil Relamation Problem... 11
2.2: Partial Echoed Listing of Empirical Input Data Variables, Waste Oil Relamation Problem... 23
2.3: Statistics, Histogram, and Graphs Produced by MOUSE for the Waste Oil Relamation Problem... 24
2.4: Schematic Diagram of the Model For The Regulatory Problem................................... 25
2.5: MOUSE Program for the Regulatory Problem.............. 27
2.6: Statistics, Histogram And Graphs Produced By MOUSE For The Regulatory Problem................. 29
3.1: General Overview of a MOUSE Program................... 30
3.2: Commented MOUSE Program For The Waste Oil Reclamation Problem.................................. 32
3.3: Commented MOUSE Program For The Regulatory Problem................................... 33
4.1: The MOUSE.DAT Data File Containing the Empirical Variable Data Lines for the Waste Oil Reclamation Problem..................... 34
4.2: Example Of A Step Rectangular Probability Distribution.. 42
4.3: Partial Data file for Fitting a Classical Distribution Using IMP............................... 44
4.4: Imp Plots Of Fitted Distibutions To Original Data... 57
4.5: Sample IMP Output, Continuous Distribution............ 59
4.6: Sample IMP Output, Discrete Distribution.............. 60
4.7: Integrating An Empirical Continuous Probability Distribution............................. 62
4.8: Input Data for an Empirical Continuous Probability Distribution............................. 64
4.9: IMP Output, Integrated Continuous Empirical Distribution, Data Of Figure 4.8.................... 65
4.10: Empirical Continuous Probability Distribution For Exercise Number 17................ 66 70

7.1: Waste Oil Reclamation Problem, Modification #1.................................... 87
7.2: Snap Output For Waste Oil Reclamation Problem, Modification #1.......................... 88
7.3: An Outline Of The MOUSE Pass.......................... 93
7.4: Partial PASS Output For Waste Oil Reclamation Problem, Modification #2.......................... 94
8.1: Regulatory Problem Modification #1.................... 98
8.2: Plot Output For Regulatory Problem, Modification #1 100
8.3: Sensitivity Output For The Regulatory Problem, Modification #1.................................... 103
9.1: Portion Of Example IMP Data File For Analysis Of A Bivariate Distribution............ 108
9.2: IMP Output, Bivariate Distribution, Data Of Figure 9.1................................. 109
9.3: Portion Of Example IMP Data File For Analysis Of A Bivariate Normal Distribution................. 115
9.4: IMP Output, Bivariate Normal Distribution Analysis, Data of Figure 9.3....................... 116
9.5: Portion Of Example IMP Data File For Analysis Of An Autocorrelation (Exponential) Time Series.. 120
9.6: IMP Output, Autocorrelated (Exponential Time Series, Data of Figure 9.5.................... 120
10.1: MOUSE Program For The Waste Oil Reclamation Problem, Modification #2............................ 122
10.2: Excerpt From Waste Oil Reclamation Problem, Modification #2, Showing Lines Relevant To Vectors, Vector Tables, And Present Value Sum Function.................................. 127
10.3: Vector Table Output For Waste Oil Reclamation Problem, Modification #2................ 127
10.4: Examples Of Skewness And Kurtosis.................... 130
10.5: Waste Oil Reclamation Problem, Modification #3.................................... 132
10.6: Output Of The Waste Oil Reclamation Problem, Modification #3.................................... 133
10.7: Plot Output For Waste Oil Reclamation Problem, Modification #2............................ 134
11.1: Original Case History................................. 142
11.2: Corrected Case History Program....................... 142
11.3: Table Output of Original Case History Program......... 144
11.4: Table Output of Corrected Case History Program.................................. 145
11.5: Placement of Trace Lines in Modification #2 of the Waste Oil Reclamation Problem............... 149

11.6: Output of The Traced Program of Figure 11.5........... 150
11.7: Output From Sample TRAP Run For Waste Oil
Reclamation Problem, Modification #2................ 157
12.1: Shapes of Some Beta Distributions..................... 173

LIST OF TABLES

Number **Page**

1.1: A Comparison of Computer Languages for Uncertainty Analysis.................................... 16
2.1: Distributions of the Input Stochastic Variables for the Irrigation Project Problem........... 20
2.2: Regulatory Levels As A Function of Delmax.............. 28
2.3: Distributions of the Input Stochastic Variables for the Regulatory Problem............................ 28
3.1: Summary of Rules for Use of Decimal Points in MOUSE... 36
12.1: Summary of Consensus Techniques............................ 171
C.1: MOUSE Control Lines................................... 185
C.2: Alphabetical List of MOUSE Interest Functions.......... 186
C.3: Alphabetical List of MOUSE Probability Functions...................................... 187
C.4: MOUSE Line Types...................................... 188

CHAPTER 1

EXPLORING UNCERTAINTY

1.1 Defining the Problem

If we define a model as a physical or symbolic representation of reality, we find among the set of all models one called the mathematical model of which one particular type consists of a series of one or more algebraic equations. Mathematical models of this type are extremely important for they are found almost everywhere, including the environment, business, economics, engineering, and science. The use of an equation is understood by almost everyone; in a somewhat "inelegant" sense, numbers "go into" the equation and an answer is obtained. For example, consider the following very simple equation,

$$Y = AB \qquad [1.1]$$

where Y might stand for an equipment purchase cost, given that A is the unit price and B is the number of units required. Alternatively, equation 1.1 might represent an engineering calculation where Y is the cross-sectional area of a heating duct, given that A is its height and B is its width. In any event, to "solve" equation 1.1 one has to know the values of A and B. If A is equal to 2, for example, and B is equal to 15, then Y = 2 x 15 = 30. Often, however, we are not sure of the values of A or of B. A might be 3 and B might be 20; in such event Y would be equal to 60, not 30. The greater our uncertainty about the input variables A and B, clearly the greater our uncertainty about the output variable, Y.

1.2 Consequences of Failure to Deal with Uncertainty

The "best" estimate of a input variable is what we state if we were restricted to single numbers for our estimate. (This is one of the single value approaches to be discussed in Section 1.3.) The use of single values (sometimes known as "point estimates") for input variables, of course, leads to single values or point estimates for output values. If we ignore uncertainty in our calculations, the point estimates obtained will almost assuredly be either too high or too low. Either may have undesirable consequences. In our equipment purchase example, an overestimate of requirements may result in overstocking the item, leading to increased inventory carrying costs. If the requirements are underestimated, downtime may occur and production schedules may not be met. In our engineering duct problem, if Y is underestimated then our heating duct will be inadequate to carry the airflow required; if Y is overestimated we will have wasted money by building it too large and the reduced air velocity might encourage the unwanted deposition of particulate matter along the duct before the air reaches the filter.

It should be pointed out that, depending upon the circumstances, overestimation may be more harmful than underestimation and vice versa. For example, it is probably better to oversize a duct rather than to undersize it since tearing out existing ductwork is very expensive. If the inventory carrying charges for an item stocked is high (an expensive, seldom-used specialized pump, for example), it may be better to understock the item and incur some occasional out-of-stock cost rather than to pay the high interest charges on unmoved inventory.

1.3 Traditional Approaches to Uncertainty

The most often encountered approaches to uncertainty in mathematical models are: (1) the best value approach, (2) the conservative approach, and (3) sensitivity analysis. The first two are single-value approaches. The "best" in "best value" is not precisely defined; generally it refers to some measure of central tendency such as an average or a mode. In a probabilistic sense, the mode is the "most likely" value in that it has the highest probability of occurring. If a probability distribution is not symmetric then the mode differs from the mean; consequently it is not always clear what another person's "best value" means. In the equipment purchase problem, we might suppose that the value of A of 2 and of B of 15 are average values. Hopefully, the answer of 30 is also some sort of average value. As we shall see in the next section, however, this is not always the case. The best value approach is an admittedly simple technique since single values are used for the input numbers to the equations. In reality, however, the best value approach is more a matter of ignoring uncertainty that it is of any conscious effort to come to grips with it.

The conservative approach does make an attempt to consider the consequences of uncertainty. As the name implies, the input values selected are not the average or most likely ones but rather those that produce conservative results with regard to the consequences of over- or underestimating. For example, in the conservative approach for the duct example, the values of A and B selected to go into equation 1.1 would be greater than their average values of 2 and 15 since, as we have suggested in the previous section, overestimation is better than underestimation in this case. If "best" values were used, there would be too great a chance that the duct area would be underestimated.

A popular traditional approach to the problem of uncertainty is sensitivity analysis. Traditional sensitivity analysis usually starts with a best value estimate, followed by a perturbation or change in one of the input variables (holding all other input variables at their previous values). The perturbation can be either an increase or a decrease in the value of the variable and hence can be either of a "conservative" nature or a "liberal" one. The perturbation is generally within the known or believed uncertainty range of the variable. The process is repeated for as many variables and for as great a change as is desired. For example, to examine the effects of a modest change in A in equation 1.1, we might increase the value of A by 10% over its "best" estimate value of 2. A 10% increase in A (to 2.2) results in an estimated Y-value of 33. If a value of A of 2.2 is "reasonably likely" to occur, then the traditional sensitivity analysis suggests that a value of Y of 33 is also "reasonably likely" to occur.

1.4 Problems with Traditional Approaches

As has been mentioned before, the best value approach really does not address the problem of uncertainty at all. Consider the three hypothetical outcomes for Y shown in Figure 1.1. In all cases the average value of Y is 30. In Case A, where there is relatively little variation in Y, there is a high probability that the actual value will be in the vicinity of the estimated (by the best value method) value of 30; in Case B, however, this likelihood is very much smaller. Estimates made by using best input values, therefore, do not necessarily have high probabilities of occurring. Note that in Case D the most likely value of Y is not the mean value of 30 but a value considerably higher. Thus, "uncertainty" is not just variance; it refers to the entire distribution, including its shape.

There is another difficulty with the best value approach which is not generally recognized. The difficulty arises when the algebraic model contains non-linear elements such as multiplications or divisions and the variables are correlated. If variables A and B of equation 1.1 were correlated, for example, the average of Y would not be equal to the average of A times the average of B. In point of fact, if A and B were positively correlated, then the average of Y would be greater than the product of the averages; conversely, if A and B were negatively correlated, the average of Y would be less than the product of the averages (see Addendum to this chapter).

The conservative approach also has its deficiencies. First, in a complex calculation involving many equations and many input variables (some of which may be correlated) it may not be obvious what values of the input variables constitute "conservative" ones with respect to the output. Second, because conservative input values generally are those with a low probability of occurring, the estimates obtained by using such values perforce will not have a high probability of occurring.

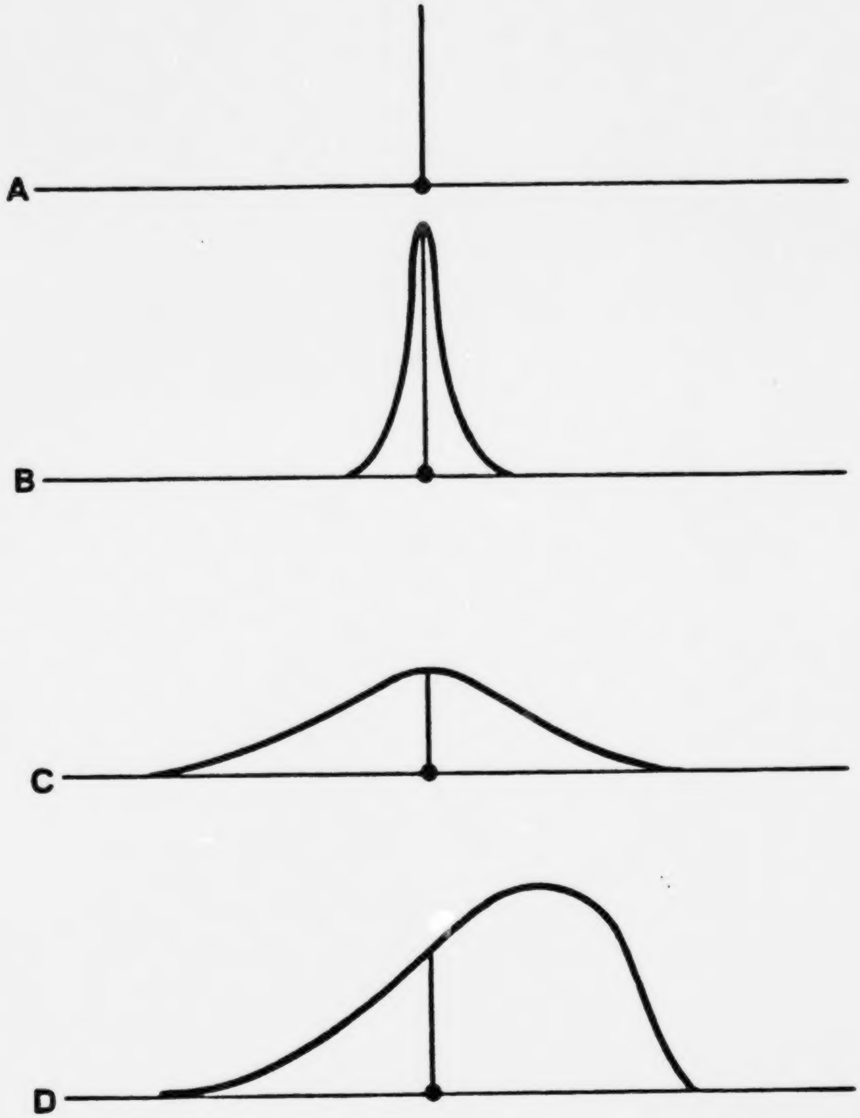


FIGURE 1.1: EXAMPLES OF UNCERTAINTY

It should be recognized that the point estimates involved in using either the best value or the conservative approach do not utilize all of the information that is usually available. One usually has at least some idea of the uncertainties in the input figures. Since a greater loss may occur in a decision from an underestimate than from an overestimate (or vice versa), there are other problems with single value approaches as well. For example, since people like success better than failure, the activities they undertake are more likely to be the ones for which they have overestimated their ability to succeed.

Traditional sensitivity analysis, being largely an amalgamation of elements of both the best value and conservative approaches, suffers the defects of both methods. An arbitrary change in the value of an input variable, even though the change falls within the expected range of the variable, tells us little about the likelihood of occurrence of the new estimate obtained. In other words, if we know little about the likelihood of such a change occurring, it follows that we know little about the likelihood of the calculated output occurring. Furthermore, in traditional sensitivity analysis all other variables are held at their previous values, the so-called "all other things being equal" view of the world. The problem is that "all other things" are seldom equal. In actuality, the change we introduce in a variable under traditional sensitivity analysis may well be either mitigated or intensified by what is happening to the other variables. In short, traditional sensitivity analysis does not show the combined net effect of changes in all variables or the likelihood of various changes occurring together. Viewed in this manner, the traditional sensitivity analysis can be misleading. More will be said about sensitivity analysis in Section 1.6.

1.5 Alternative Solutions to the Uncertainty Problem

Let us now examine three alternative solutions to the problem of uncertainty. The first, illustrated in Figure 1.2, is Direct or Complete Enumeration. The model of equation 1.1 is employed, and we assume the uncertainties of A and B as given in the two probability distributions for these variables shown in the upper left-hand corner of the figure. In other words, we suppose that there is a 25% chance that A is equal to 1, 50% that it is equal to 2, and 25% that it is equal to 3. For B, there is a 50-50 chance that it is equal to either 10 or 20. (For this simple example, we assume no correlation between the two input variables.) In complete enumeration we list all of the possible combinations of the input variables and then calculate the probabilities of these combinations occurring. In this example there are 3 choices for A and 2 for B, resulting in 6 possible outcomes for Y. The probabilities of these combinations are shown in the middle top of the figure. Since some of the combinations are duplications, the table of combinations of A and B may be simpli-

MODEL: $Y = AB$

A	p(A)	B	p(B)
1	.25	10	.50
2	.50	20	.50
3	.25		

A	B	AB	p(AB)
1	10	10	.125
2	10	20	.250
3	10	30	.125
1	20	20	.125
2	20	40	.250
3	20	60	.125

AB	p(AB)
10	.125
20	.375
30	.125
40	.250
60	.125

Note: Average = $\overline{Y} = \bar{A}\bar{B} = 2(15) = 30$

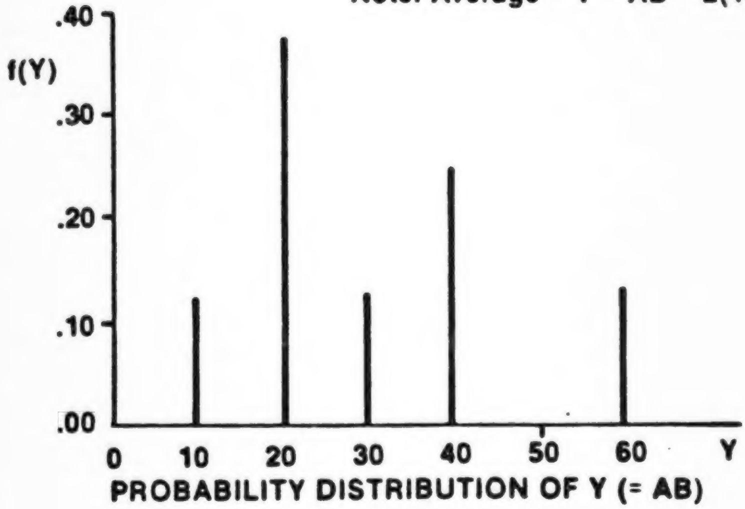


FIGURE 1.2: DIRECT (COMPLETE) ENUMERATION

fied to the 5 entries shown at the upper right of the figure. The average value of Y is shown to be 30. At the bottom of Figure 1.2 is a graph of the frequency or probability distribution of Y. Note that the most likely value is not the average but rather values to either side of it. Furthermore, one of the extreme values (Y = 60) has a higher probability of occurrence than has the average value. As can be seen, the complete enumeration method tells us everything about the distribution of Y, including its mean, standard deviation, minimum, maximum, and the probability of occurrence of any value of Y.

The second method we shall consider is the Probability Calculus method. The method, as the name implies, requires some knowledge of the calculus of probabilities (sometimes known in engineering as the "propagation of error".... see Deming, 1950). Using the model of equation 1.1 as before, the method is illustrated in Figure 1.3. The error formula is given at the top of the figure, and involves three terms and knowledge of the variances of A and B. The latter are calculated, as is shown in the figure, from the probability distributions of A and B given previously in Figure 1.2. The error formula shows the variance of Y to be 225. The Probability Calculus method produces no more than the mean and the variance of the output (i.e., Y) distribution. That the variance alone is not sufficient to determine the nature of uncertainty is clear from Figure 1.1. It is entirely possible that Cases B and C of Figure 1.1 have the same variance; they may have, however, quite different consequences to the decision maker.

The third approach to the problem of uncertainty is a form of Monte Carlo simulation known as Model Sampling. The term "Monte Carlo" originated during World War II where it referred to a method, utilizing random numbers, used for solving certain problems where the passage of time played no substantive role. The idea of Model Sampling is relatively simple:

a. A value for each of the input variables is drawn at random from their respective probability distributions, and the model is computed using this particular set of values.

b. The above process is repeated many times. Since the results vary with each iteration, the outputs themselves (i.e., the Y's) are gathered in the form of a probability distribution. Thus the uncertainties of the model's inputs are transferred to the output which can then be studied and subsequently utilized in decision processes.

MODEL: $Y = AB$

ERROR FORMULA IS $\mathrm{var}\,(AB) = \mathrm{var}\,(Y) = \bar{A}^2\,\mathrm{var}\,(B) + \bar{B}^2\,\mathrm{var}\,(A) + \mathrm{var}\,(A)\,\mathrm{var}\,(B)$

$\bar{A} = 2$ and $\bar{B} = 15$

$p(A)$	A	$(A - \bar{A})^2$	$p(A)\,(A - \bar{A})^2$
.25	1	1	.25
.50	2	0	.00
.25	3	1	.25
		var(A) =	.50

$p\,(B)$	B	$(B - \bar{B})^2$	$p(B)\,(B - \bar{B})^2$
.50	10	25	12.5
.50	20	25	12.5
		var (B) =	25.0

therefore, $\mathrm{var}\,(Y) = 2^2\,(25.0) + 15^2\,(0.50) + (0.50)(25.0) = 225$

FIGURE 1.3: PROBABILITY CALCULUS APPROACH

The procedure is shown schematically in Figure 1.4. The output (and the accuracy) of the Monte Carlo simulation method becomes almost equal to that of complete enumeration as the number of iterations becomes large. The Monte Carlo simulation method is the one used in this manual and forms the basis for MOUSE, a computerized uncertainty analysis system which is the subject of this manual.

1.6 Traditional Sensitivity and Uncertainty Sensitivity Analysis Compared

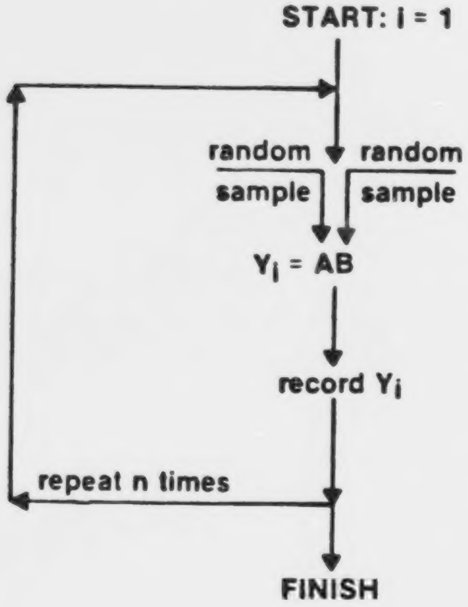
A pictorial comparison of traditional sensitivity analysis with uncertainty sensitivity analysis is shown in Figures 1.5 and 1.6. In traditional sensitivity analysis we alter the location (i.e., value) of one variable (holding all others at their previous values) and examine the effect on the output variable (Figure 1.5). This is clearly a two-step procedure (T1 and T2). In uncertainty sensitivity analysis (Figure 1.6) this is a one-step procedure (U1) since all perturbations in X are automatically reflected in the distribution of Y. The perturbations in the X-values, unlike those in traditional sensitivity analysis, are not arbitrary, however; they are performed in accordance with their likelihoods of occurring.

In uncertainty sensitivity analysis we can not only alter the value of an input variable, but we can alter its location (U2 in Figure 6), the amount of its uncertainty (U3) and its nature (U4). The U3 pair of distributions in Figure 1.6 shows one possible result. By decreasing the variability of the input variable by the amount shown, we decrease our uncertainty about the output by a fairly large amount. The implication is that it would pay us to reduce the uncertainty in this input variable, perhaps by obtaining better data. The decrease in the uncertainty of the output variable could have been much less. If this were so, we would have concluded that it would not have paid to invest additional money in reducing the uncertainty of the input variable. The possibilities for sensitivity analysis using the uncertainty approach are therefore, far greater than those available using any single value, traditional approach.

1.7 Languages For A Computerized Uncertainty Analysis System

Monte Carlo simulation requires the use of a digital computer for any substantive problem. It is important, therefore, to consider the desirable characteristics of any computerized uncertainty analysis system:

METHOD 3: MONTE CARLO SIMULATION


START: I = 1
random
random
sample
sample
$Y_i = AB$
record Y_i
repeat n times
FINISH

FROM COLLECTION OF Y's OBTAIN:

1. **MEAN**
2. **STANDARD DEVIATION**
3. **COEFFICIENT OF VARIATION**
4. **MINIMUM**
5. **MAXIMUM**
6. **GRAPH OF FREQUENCY DISTRIBUTION**
7. **GRAPH OF CUMULATIVE FREQUENCY DISTRIBUTION**

FIGURE 1.4: MONTE CARLO SIMULATION

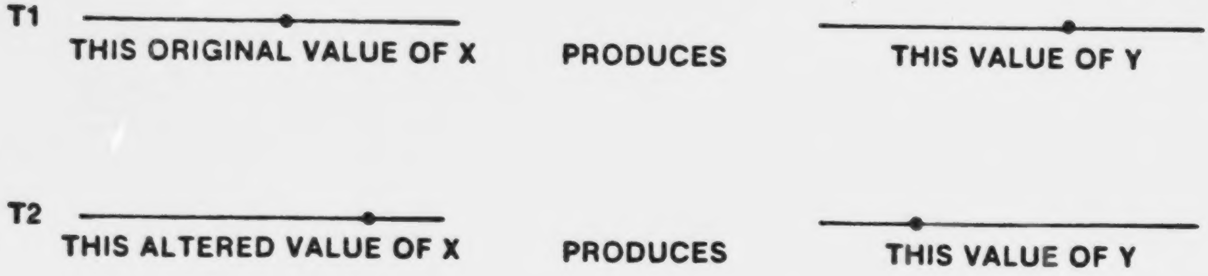

T1
THIS ORIGINAL VALUE OF X
PRODUCES
THIS VALUE OF Y
T2
THIS ALTERED VALUE OF X
PRODUCES
THIS VALUE OF Y

FIGURE 1.5: SENSITIVITY ANALYSIS: TRADITIONAL METHOD

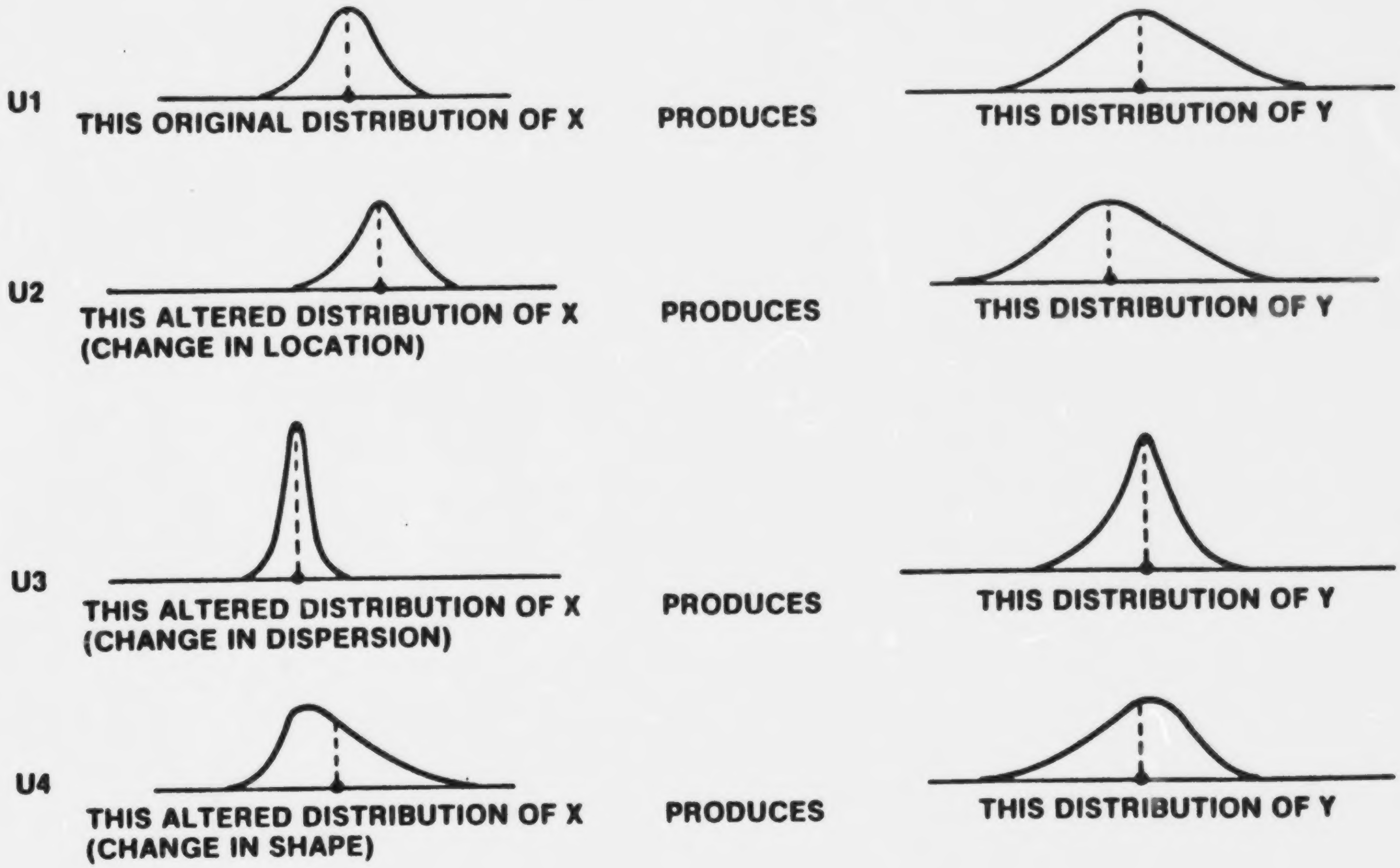

U1
THIS ORIGINAL DISTRIBUTION OF X
PRODUCES
THIS DISTRIBUTION OF Y
U2
THIS ALTERED DISTRIBUTION OF X (CHANGE IN LOCATION)
PRODUCES
THIS DISTRIBUTION OF Y
U3
THIS ALTERED DISTRIBUTION OF X (CHANGE IN DISPERSION)
PRODUCES
THIS DISTRIBUTION OF Y
U4
THIS ALTERED DISTRIBUTION OF X (CHANGE IN SHAPE)
PRODUCES
THIS DISTRIBUTION OF Y

FIGURE 1.6: SENSITIVITY ANALYSIS: UNCERTAINTY METHOD

a. Simplicity

In order to conduct Monte Carlo simulation upon an environmental, business, economic, engineering or scientific model, you must communicate the model and any input information to the computer, and to instruct it as to what is required with regard to the nature and the output of the analysis. Naturally enough, this requires a large number of tasks to be accomplished by the computer. Obviously, it would be highly desirable if you need not have prior knowledge of computer programming or even of computers in general to use such a system. Furthermore, the programming language used should be simple so that you can spend his time improving his model and his probability estimates rather than worrying about the details of programming.

b. Library

An important part of uncertainty analysis is the probability distributions involved and, if economic matters form part of the model, the interest functions used. A desirable feature of any computerized uncertainty analysis system, then, would be a large library of probability and interest functions.

c. Automatic Output Distributions

The major end product of any uncertainty analysis consists of the probability distributions of the quantities we are trying to estimate as well as various statistics concerning these distributions. It is essential, therefore, that any uncertainty analysis system provide automatic output of these critical elements.

d. Intermediate Output Distributions

In any complex model, the analyst is often interested in intermediate outputs as well. In an equipment expansion problem, for example, in addition to the total cost both capital and operating costs are also of great concern. The uncertainty analysis system should be able to describe the uncertainties in any of these intermediate outputs.

e. Generalized Sensitivity Analysis

In Section 1.6 the value of more generalized sensitivity analysis was discussed. The uncertainty analysis system should facilitate such types of investigations.

f. De-bugging Tools

It is easy to lose one's way in a complex model. Errors in such models are often not easy to find. An answer may look reasonable but the model or its implementation may still be faulty. For small models a hand calculation, line-by-line, is one way to uncover such errors but for large models this approach is slow and prone to error. A desirable feature of an uncertainty analysis system, then, is some painless way to "debug" the program.

g. Correlated Variables

It is true that calculations are much simpler when one assumes that all of the input variables are uncorrelated. Unfortunately, this rarely occurs in the real world. Supply varies with price, yield varies with pressure, and sales vary with the season. Any uncertainty analysis system should be able to accommodate correlated variables, i.e., correlation of more than just the "A = 0.7B" kind.

h. Derivation of Probability Distributions

A critical aspect of uncertainty analysis is the determination of the probability distributions of the input variables. These may be determined subjectively, by fitting empirical data to classical probability distributions or by a combination of these methods. To newcomers to uncertainty analysis this is often a more formidable task than the construction of the model itself. A truly useful uncertainty analysis system should provide some means for helping you derive the probability distributions required.

i. Modular Expansion

At times you may wish to modify an uncertainty analysis system to meet special requirements of his own particular model, e.g., the addition of a probability distribution not included in the original system. Modular construction of such systems facilitates such "tailoring".

It is possible to use general purpose computer languages such as BASIC, FORTRAN or PASCAL for uncertainty analysis but the problems are formidable. For one thing, specialized programming knowledge is required and these languages are not mastered without some effort. There is nothing particularly simple about either FORTRAN or PASCAL as those who have had to deal with the mysteries of the FORMAT statement in the former can attest. There is no such thing as "automatic output" or provision for sensitivity analysis. If you want these features you must painstakingly program them himself. They are, however, transportable in that compilers or translators are available for these languages for almost every type of computer.

Another alternative is the specialized computer simulation programs, foremost among them being GPSS, SIMSCRIPT, and SLAM. (For references to these simulation languages see Maryanski, 1980 and Pritsker & Pegden, 1979.) One major difficulty with GPSS is that, although easy to learn, it is based upon integer arithmetic, a decidedly unhealthy environment for the kind of calculations needed for uncertainty analysis. GPSS was primarily designed for job-shop problems and is not appropriate to uncertainty analysis. SIMSCRIPT, on the other hand, although perfectly capable of performing uncertainty analysis, is a complex language usually taught as a hierarchal set of languages: first, as a scientific language similar to FORTRAN; then as a general purpose language comparable to ALGOL; next as a list processing language to aid in the creation and handling of data structures; and finally as a simulator and statistics gathering language. SLAM is a collection of FORTRAN functions and subroutines that you must assemble having first a knowledge of FORTRAN.

There are three general approaches taken by general purpose simulation languages: they are either event-oriented, process-oriented, equation-oriented or some combination of the three. In event-orientation, a system is modeled by identifying its characterizing events and then writing a set of event routines that describe how the state of the system changes whenever an event occurs. In a simple queuing system, for example, the two major events are the arrival of a unit to be serviced and the completion of service of a unit. In process-orientation, the model is constructed about entities that are "processed" as they move through the system being modeled. In the simple queuing system, for example, a unit enters and leaves a queue, enters a service station, receives service, leaves the service station, etc. Most general purpose simulation languages are event- or process-oriented and, accordingly, contain features such as control of time, set membership, and queuing routines not needed for dealing with sets of equations.

Table 1.1 presents a comparison among general purpose computer languages, general purpose simulation languages, and MOUSE (the system introduced in this manual) for uncertainty problems that deal in sets of algebraic equations. Table 1.1 is not designed to devalue the capabilities of either general purpose computer languages or general purpose simulation languages. MOUSE is a restricted special purpose simulation language and the table makes its comparison assuming that the problem at hand is of this restricted form. (However, see Exercise No. 8, Chapter 10, for an example of MOUSE used as a more general purpose simulation language.) For such restricted models, MOUSE is clearly superior to the other two groups of languages. It is concise, powerful, and convenient to use. Whether you are a scientist, engineer, businessman or economist, MOUSE can solve uncertainty problems faster and easier than can other languages. With MOUSE, your attention is on problem-solving, rather than on the details of coding a program to compute a solution. Further, MOUSE programs are easier to understand, easier to explain to others, and easier to modify than are general purpose languages.

ADDENDUM

In Section 1.4 it was mentioned that, for non-linear models, the use of average values as inputs does not guarantee that the output is an average value. For example, consider the quantity A^2, with A having the probability distribution shown in Figure 1.2. The average or "expectation" of A, E[A], is shown in Figure 1.2 to be 2. It might be thought that, since A^2 = A times A, the average of A^2 (i.e., $E[A^2]$) is E[A]E[A] or 2(2) = 4. This is not the case, however, since $E[A^2]$ is easily seen to be:

$$1(0.25) + 4(0.50) + 9(0.25) = 4.50.$$

The expectation of a product in general is (Kane, 1968):

$$E[AB] = E[A]\ E[B] + COV(A,B) \qquad [1.2]$$

where COV(A,B) is the covariance between A and B. Thus,

$$E[A^2] = E[A]\ E[A] + COV(A,A). \qquad [1.3]$$

Since COV(A,A) = VAR(A) (i.e., the variance of A) and noting from Figure 1.3 that VAR(A) = 0.50, using equation 1.3 we obtain:

$$E[A^2] = 2(2) + 0.50 = 4.50$$

i.e., the answer previously obtained.

If A and B are independent, then COV(A,B) = 0 and E[AB] = E[A]E[B]. In equation [1.1], A and B were selected independently of one another; accordingly, E[Y] = E[AB] = E[A]E[B] or 2(15) = 30. Note, however, that

$$COV(A,B) = rho[VAR(A)\ VAR(B)]^{0.5} \qquad [1.4]$$

where rho is the correlation coefficient between A and B. From Figure 1.3, VAR(B) = 25.0. If A and B were correlated to the extent that rho = 0.5, then from equations 1.2 and 1.4:

$$E[AB] = 2(15) + 0.5[(0.50)(25.0)]^{0.5} = 31.77$$

If the correlation were negative, i.e., rho = -0.50, then

$$E[AB] = 2(15) - 0.5[(0.50)(25.0)]^{0.5} = 28.23$$

TABLE 1.1

A COMPARISON OF COMPUTER LANGUAGES FOR UNCERTAINTY ANALYSIS

\ Language \ \ Element \	General Purpose Languages (e.g., FORTRAN, PASCAL, BASIC)	General Purpose Simulation Languages			MOUSE
		GPSS	SIMSCRIPT	SLAM	
Simplicity	complex	simple	complex	complex	simple
Built-in Probability functions	no	a few	yes	yes	yes
Built-in Interest functions	no	no	no	no	yes
Automatic Output	no	yes	yes	yes	yes
Decimal Numbers	yes	no	yes	yes	yes
Automatic Sensitivity Analysis	no	no	no	no	yes
Correlation Among Variables	must be programmed	no	must be programmed	must be programmed	yes
De-bugging Simple	no	no	no	no	yes
Aid To Deriving Probability Distributions	no	no	no	no	yes*
Ability to Expand by Modules	yes	no	no	yes	yes

*With its companion system, IMP. See Chapters 12 and 13.

REFERENCES

Deming, W.E., "Some Variances in Random Sampling", in Some Theories of Sampling, John Wiley & Sons, New York, N.Y., p. 127-134,1950.

Kane, E. J., Economic Statistics and Econometrics, Harper & Row, New York, N.Y., 1968.

Maryanski, F. J., Digital Computer Simulation, Hayden Book Co., Rochelle Park, N.J., 1980.

Pritsker, A., and Pegden, C. D., Introduction To Simulation and Slam, John Wiley & Sons, New York, N.Y., 1979.

CHAPTER 2

EXAMPLES OF UNCERTAINTY ANALYSIS

2.1 A Cost Example

This is a simple cost example that involves a waste reclamation project where the present value of the returns is a function of the cost of establishing the system, which involves the re-refining of spent motor oil, and of a number of annual equal net returns after the project is completed. The annual net returns for this project consist of revenues obtained from the sale of three reclaimed petroleum products, and can be calculated from a series of algebraic equations as follows:

a. The throughput in production of motor oil grade 1, THRU1, is a function of the contaminants of the feed, CONTAM, i.e.,

THRU1 = 10 - CONTAM

In other words, as the level of contaminants increases, less of motor oil grade 1 will be produced.

b. The yield of motor oil grade 1, YIELD1, is also a function of this same contaminant level, i.e.,

YIELD1 = 10 - 2*CONTAM

c. The production of motor oil grade 1, PROD1, is throughput, THRU1, times yield, YIELD1, plus a random effect, RAND1, i.e.,

PROD1 = THRU1*YIELD1 + RAND1

This random effect can either be positive or negative, and represents uncertainties of the demand for motor oil grade 1.

d. The gross revenue from motor oil grade 1, GREV1, is its price, PRICE1, times the quantity produced, PROD1, i.e.,

GREV1 = PRICE1*PROD1

e. The price of motor oil grade 2, another product of the reclaiming facility, is designated by PRICE2. This price is dependent upon both the production of motor oil grade 2, PROD2, and another random effect, RAND2, i.e., PRICE2 = 10 - SLOPE*PROD2 + RAND2

SLOPE is part of the functional relationship between price and production in this equation, and RAND2 represents the uncertainties of the marketplace.

f. The gross revenue from motor oil grade 2, GREV2, is price, PRICE2, times production, PROD2, i.e.,

GREV2 = PRICE2*PROD2

g. The gross revenue from motor oil grade 3, the third and last product produced at the facility, is price, PRICE3, times production, PROD3, i.e.,

GREV3 = PRICE3*PROD3

h. The annual net benefit, ANETB, is the sum of the gross revenues, GREV1, GREV2, and GREV3, less the annual cost (assumed to be fixed in this problem and equal to 70), i.e.,

ANETB = GREV1 + GREV2 + GREV3 - 70

i. The present value, PVALUE, of the returns is a function of the life of the investment, XLIFE, the interest rate (which is assumed in this problem to be a constant and equal to 0.08), the annual benefits, ANETB, and the initial investment, XININV, i.e.,

$$\text{PVALUE} = \text{ANETB} \sum_{t=1}^{\text{XLIFE}} (1 + 0.08)-t - \text{XININV}$$

These nine equations comprise the mathematical model.

2.2 The Input Variables

Ten of the input variables in this model are assumed to represent uncertain quantities. Two of them, RAND1 and RAND2, have already been mentioned. However, our analyst is also uncertain about the values of CONTAM, PRICE1, the SLOPE coefficient in the PRICE2 equation, PROD2, PRICE3, PROD3, XLIFE, and XININV. Each variable is assumed to have a probability distribution; in the language of probability they are termed stochastic input variables. The problem of determining the probability distribution of stochastic input variables is left to Chapters 12 and 13. For the time being, however, let us assume that the analyst has determined these distributions to be those shown in Table 2.1.

TABLE 2.1

DISTRIBUTIONS OF THE INPUT STOCHASTIC VARIABLES FOR THE WASTE OIL RECLAMATION PROBLEM

Variable	Type	Outcome	Probability	Cumulative Probability
CONTAM	Discrete Uniform	1	.25	-
		2	.25	-
		3	.25	-
		4	.25	-
RAND1	Continuous Uniform	-10	-	.00
		10	-	1.00
PRICE1	Discrete Empirical	7.5	.35	.35
		8.0	.40	.75
		10.5	.25	1.00
SLOPE	Continuous Empirical	.05	-	.00
		.10	-	.30
		.15	-	.70
		.20	-	1.00
PROD2	Step Rectangular	30 to 35	.15	.15
		35 to 60	.50	.65
		60 to 65	.35	1.00

Variable	Type	Parameters
RAND2	Normal	0.00 (= mean) 0.10 (= standard deviation)
PRICE3	Empirical Beta	3.5 (= lowest value) 5.0 (= most likely value) 6.5 (= highest value)
PROD3	Exponential	18 (= mean)
XLIFE	Discrete Triangular	5 (= lowest value) 10 (= most likely value) 15 (= highest value)
XININV	Trapezoidal	1600 (= lowest value) 2000 (= most likely value, lower boundary) 2200 (= most likely value, upper boundary) 2400 (= highest value)

Table 2.1 shows, for example, that variable CONTAM has four possible outcomes (representing four different levels of contaminants): 1, 2, 3, and 4, each with probabilities 0.25. Detailed descriptions of all of the probability distributions used can be found in Chapter 4.

2.3 The "Best Value" Approach

In the traditional best value approach, the average values of the input variables are entered into the mathematical model as follows:

CONTAM = 2.50000
RAND1 = .00000
PRICE1 = 8.45000
SLOPE = .12500
PROD2 = 52.50000
RAND2 = .00000
PRICE3 = 8.00000
PROD3 = 18.00000
XLIFE = 10.00000
XININV = 2100.00000

The intermediate and final equations evaluated at these values are:

THRU1 = 7.50000
YIELD1 = 5.00000
PROD1 = 37.50000
GREV1 = 316.87500
PRICE2 = 3.43750
GREV2 = 180.46880
GREV3 = 144.00000
ANETB = 571.34380
PVALUE = 1733.76300

As we shall see shortly, 1733.76 is not the correct value for the present worth of this project.

2.4 The MOUSE Oil Reclamation Program And Results

Figure 2.1 shows the MOUSE program required for the Waste Oil Reclamation problem. There are 43 lines of computer code but only the following nine lines were written by the user directly:

```
THRU1=10.0-CONTAM
YIELD1=10.0-2.0*CONTAM
PROD1=THRU1*YIELD1+RAND1
GREV1=PRICE1*PROD1
PRICE2=10.0-SLOPE*PROD2+RAND2
GREV2=PRICE2*PROD2
GREV3=PRICE3*PROD3
ANETB=GREV1+GREV2+GREV3-70.0
PVALUE=ANETB*UPW(0.08,XLIFE,0.0)-XININV
```

since a MOUSE utility, called "TRAP" (see Chapter 11), wrote (with some user input) the remaining lines. A full explanation for the MOUSE program of Figure 2.1 will be given in the Chapters that follow. The output for this program is shown in Figures 2.2 and 2.3. Figure 2.2 shows a portion of the automatically echoed listing of the empirical input variables data. Figure 2.3 shows the statistics, histogram and graph produced. The quantity tabled in Figure 2.3 is PVALUE, to which the table name "PRES VAL" has been given. The number of iterations (as described in Chapter 1, Section 1.5), mean, standard deviation, coefficient of variation, average of values less than zero, minimum, and maximum of PVALUE is part of this standard output, as is the histogram table containing the number of entries, percent entries, cumulative percent entries, and complement of the cumulative percent entries for each interval of the histogram. Graphical output is also automatically provided in the form of (a) the frequency distribution of PVALUE (asterisks), and (b) the cumulative frequency distribution of PVALUE (circles). Note that the true mean is estimated as 581.57. The "best" value approach as calculated in the previous section resulted in a value approximately 50% greater. This is due to the non-linear functional relationships reflected in the uncertainty equations. The exact mean present value, obtained by applying mathematical expectations, is 574.

2.5 A Regulatory Example

This example concerns the development of a means to establish the regulatory level of those wastes disposed of in a landfill that pose a hazard due to their toxic organic constituents. The specific environment modeled was that of a non-secure sanitary landfill that receives a small amount of industrial waste, the source of the toxic materials. The specific features of the model are as follows:

(1) The landfill receives predominately domestic refuse, i.e., only 5% of the waste is industrial in nature;

```
C
C THE WASTE OIL RECLAMATION PROBLEM
C
      DIMENSION D1(3,2),D2(4,2),D3(4,2)
      DIMENSION TAB(46)
      COMMON/Q1/C80/Q2/JUMP/Q3/LOOP/Q4/SENS/Q5/IDATA/Q6/IOUT/Q7/IPASS/Q8
     &/IRAN/Q9/IREAD/Q10/ISCR1/Q11/ISCR2/Q12/ISNAP/Q13/ITER/Q14/ITOG/Q15
     &/IWRIT/Q16/IXXX/Q17/OK1/Q18/OK2/Q19/OK3/Q20/OK4/Q21/OK5/Q22/OK6
      CALL NAME('(WASTE OIL RECLAMATION PROBLEM)')
      CALL READ(D1,3)
      CALL READ(D2,4)
      CALL READ(D3,4)
      ITER=5000
5     DO 10 IXXX=1,ITER
      CONTAM=DUNI(1.0,4.0)
      RAND1=CUNI(-10.0,10.0)
      PRICE1=EDIS(D1,3)
      SLOPE=ECTS(D2,4)
      PROD2=STEP(D3,4)
      RAND2=GAU(0.00,0.10)
      PRICE3=EBETA(3.5,5.0,6.5)
      PROD3=EXPON(18.0)
      XLIFE=DTRI(5.0,10.0,15.0)
      XININV=TRA(1600.0,2000.0,2200.0,2400.0)
      THRU1=10.0-CONTAM
      YIELD1=10.0-2.0*CONTAM
      PROD1=THRU1*YIELD1+RAND1
      GREV1=PRICE1*PROD1
      PRICE2=10.0-SLOPE*PROD2+RAND2
      GREV2=PRICE2*PROD2
      GREV3=PRICE3*PROD3
      ANETB=GREV1+GREV2+GREV3-70.0
      PVALUE=ANETB*UPW(0.08,XLIFE,0.0)-XININV
      CALL TABLE(PVALUE,'(PRES VAL)',DUM0,DUM1,40,TAB)
10    CONTINUE
      GOTO(5,15), JUMP
15    CONTINUE
      END
```

FIGURE 2.1

MOUSE PROGRAM FOR THE WASTE OIL RECLAMATION PROBLEM

```
                    ********************
                    *    INPUT DATA    *
                    ********************


INPUT DATA DISTRIBUTION FOR
PRICE1         (PRICE OF MOTOR OIL GRADE 1)

VALUE OF VARIABLE         CUMULATIVE PROBABILITY


         7.50000                       .35000
         8.00000                       .75000
        10.50000                      1.00000




INPUT DATA DISTRIBUTION FOR
SLOPE          (SLOPE OF MOTOR OIL GRADE 2 PRICE EQUATION)

VALUE OF VARIABLE         CUMULATIVE PROBABILITY


          .05000                       .00000
          .10000                       .30000
          .15000                       .70000
          .20000                      1.00000




INPUT DATA DISTRIBUTION FOR
PROD2          (PRODUCTION OF MOTOR OIL GRADE 2)

VALUE OF VARIABLE         CUMULATIVE PROBABILITY


        30.00000                       .00000
        35.00000                       .15000
        60.00000                       .65000
        65.00000                      1.00000
```

FIGURE 2.2:

PARTIAL ECHOED LISTING OF EMPIRICAL INPUT DATA VARIABLES,

WASTE OIL RECLAMATION PROBLEM

```
                    ********************************************
                    *    DISTRIBUTION FOR QUANTITY PRES VAL    *
                    ********************************************


NUMBER OF ITERATIONS =  5000          MEAN =      1841.81500          STANDARD DEVIATION =       1942.75900
                                   MINIMUM =     -3190.54000   COEFFICIENT OF VARIATION, % =       105.48070
                                   MAXIMUM =     11204.57000

AVERAGE OF VALUES LESS THAN ZERO =       -782.08850


         LOWER  NUMBER OF  PERCENT  CUMULATIVE  CUMULATIVE                 DISTRIBUTIONS
         LIMIT    ENTRIES  ENTRIES   % ENTRIES  COMPLEMENT    * = FREQUENCY DISTRIBUTION     0 = CUMULATIVE DISTRIBUTION
    -3380.0000         1.      .02         .02       99.98   0
    -3040.0000         4.      .08         .10       99.90   0*
    -2700.0000        12.      .24         .34       99.66   0**
    -2360.0000        27.      .54         .88       99.12   *0***
    -2020.0000        44.      .88        1.76       98.24   *0******
    -1680.0000        70.     1.40        3.16       96.84   **0**********
    -1340.0000        91.     1.82        4.98       95.02   ***0************
    -1000.0000       176.     3.52        8.50       91.50   *****0************************
     -660.0000       203.     4.06       12.56       87.44   ********0**************************
     -320.0000       243.     4.86       17.42       82.58   ***********0*****************************
       20.0000       310.     6.20       23.62       76.38   **************0*************************************
      360.0000       325.     6.50       30.12       69.88   ******************0************************************
      700.0000       369.     7.38       37.50       62.50   ***********************0*************************************
     1040.0000       339.     6.78       44.28       55.72   ***************************0*****************************
     1380.0000       327.     6.54       50.82       49.18   *******************************0***********************
     1720.0000       346.     6.92       57.74       42.26   ***********************************0**********************
     2060.0000       285.     5.70       63.44       36.56   ***************************************0********
     2400.0000       308.     6.16       69.60       30.40   ******************************************0*********
     2740.0000       248.     4.96       74.56       25.44   ******************************************   0
     3080.0000       251.     5.02       79.58       20.42   ******************************************       0
     3420.0000       222.     4.44       84.02       15.98   **************************************             0
     3760.0000       167.     3.34       87.36       12.64   *****************************                        0
     4100.0000       152.     3.04       90.40        9.60   **************************                             0
     4440.0000       114.     2.28       92.68        7.32   ********************                                     0
     4780.0000        81.     1.62       94.30        5.70   **************                                            0
     5120.0000        75.     1.50       95.80        4.20   *************                                             0
     5460.0000        60.     1.20       97.00        3.00   ***********                                                0
     5800.0000        41.      .82       97.82        2.18   ********                                                    0
     6140.0000        25.      .50       98.32        1.68   *****                                                       0
     6480.0000        22.      .44       98.76        1.24   *****                                                       0
     6820.0000        21.      .42       99.18         .82   ****                                                        0
     7160.0000        10.      .20       99.38         .62   ***
     7500.0000         8.      .16       99.54         .46   **
     7840.0000         6.      .12       99.66         .34   **
     8180.0000         7.      .14       99.80         .20   **
      OVERFLOW        10.      .20      100.00         .00   ***


CUMULATIVE  CUMULATIVE        VALUE OF
 % ENTRIES  COMPLEMENT        PRES VAL


        .1        99.9      -3040.0000
       1.0        99.0      -2313.6360
       5.0        95.0      -1338.0680
      10.0        90.0       -874.3843
      25.0        75.0         92.1846
      50.0        50.0       1337.3700
      75.0        25.0       2769.8010
      90.0        10.0       4055.2630
      95.0         5.0       4938.6660
      99.0         1.0       6674.2850
```

FIGURE 2.3

STATISTICS, HISTOGRAM AND GRAPHS PRODUCED

BY MOUSE FOR THE WASTE OIL RECLAMATION PROBLEM

(2) The character of the leaching fluid that the waste will be exposed to is predominately a function of the non-industrial material in the landfill;

(3) The landfill is situated over an aquifer that is a source of drinking water;

(4) The soil below the site is composed of material with limited attenuation capacity;

(5) Persons using the aquifer as a source of drinking water are supplied from wells situated 150 meters down-gradient from the landfill;

(6) As constituents migrate from the landfill through the saturated and unsaturated zones to the source of the drinking water, they are attenuated by a factor of 100.

A schematic of the model is shown in Figure 2.4.

The regulatory level (REGLEV) of a toxic constituent is obtained by simply multiplying the maximum permitted daily exposure level (DELMAX) for the toxic constituent in question by a suitable factor, i.e.,

REGLEV(ppm) = DELMAX(mg/day) x FACTOR(days/kg)

since mg/kg is equivalent to ppm. The factor can be explained by the following dimensional analysis:

FACTOR = [(attenuation factor) x (liters leached/year) x years]/
[(liters consumed/day) x (industrial waste quantity, kg)].

If the (dimensionless) attenuation factor is assumed to be 100, and if we assume 2 liters of water consumption per day, a 70-year exposure period, and that 5% of the waste is industrial, the factor becomes:

FACTOR = 3500 * [liters leached/year]/

[industrial waste quantity, kg]. =

[annual precipitation, liters/sq. meters per year x

rate of leaching (as a fraction of the annual precipitation)]/

[landfill depth, meters x waste density, kg/cubic meters x 0.05].

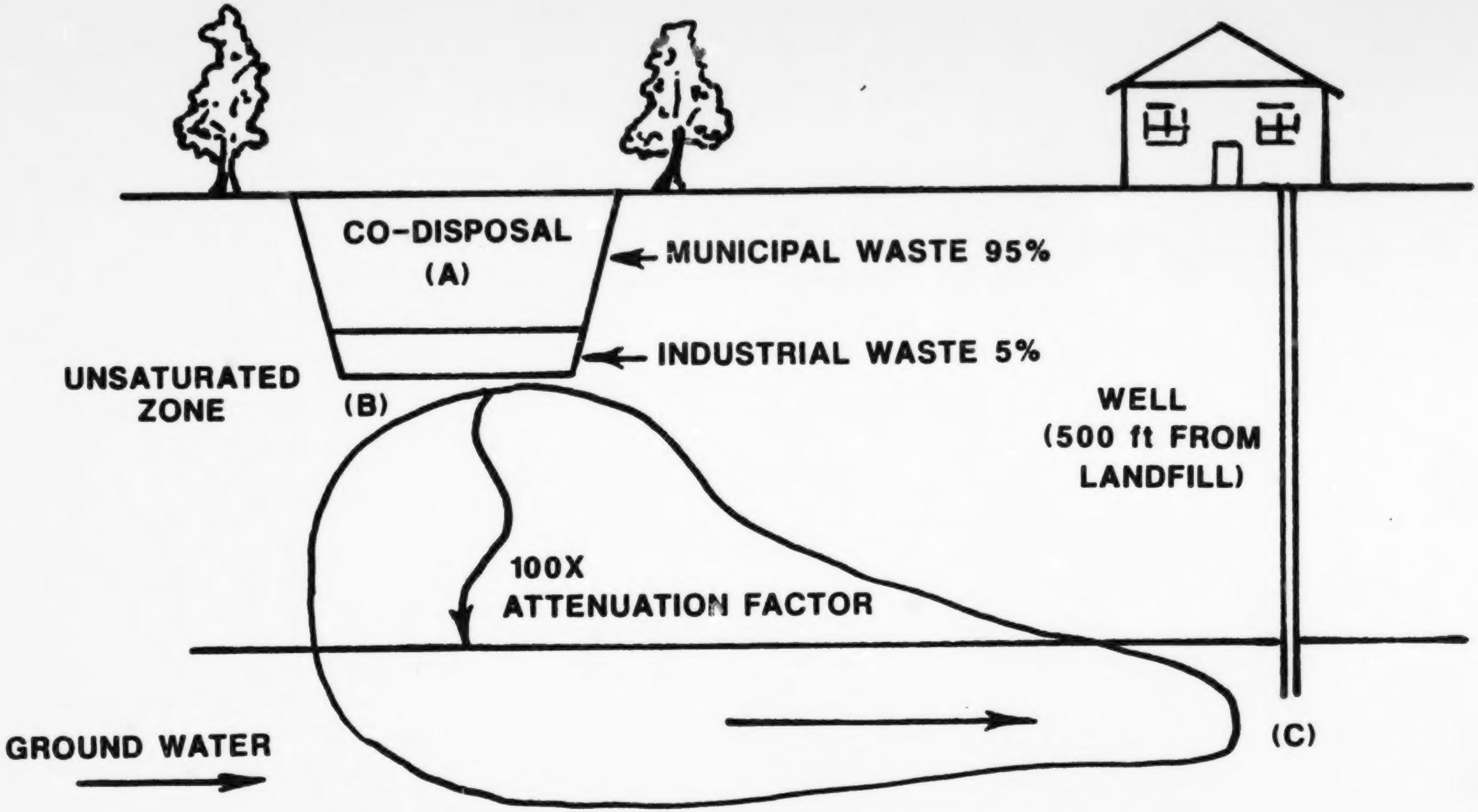

CO-DISPOSAL
(A)
MUNICIPAL WASTE 95%
INDUSTRIAL WASTE 5%
UNSATURATED ZONE
(B)
WELL
(500 ft FROM LANDFILL)
100X
ATTENUATION FACTOR
GROUND WATER
(C)

FIGURE 2.4

SCHEMATIC DIAGRAM OF THE MODEL FOR THE REGULATORY PROBLEM

Letting annual precipitation = PRECIP, rate of leaching = RLEACH, landfill depth = DEPTHL, and waste density = WDENS, then

$$FACTOR = 70{,}000 * [PRECIP * RLEACH]/[DEPTHL * WDENS].$$

Thus, the regulatory level of a toxic constituent in this model is a function of four factors: annual precipitation, rate of leaching, landfill depth, and waste density. Suppose the annual precipitation was 1000 liters/meter per year, the rate of leaching was 0.8, the landfill depth was 8 meters, and the waste density was 500 kg/cub. meter. Then FACTOR would equal 14000. At this value of FACTOR, Table 2.2 shows the regulatory levels for compounds with varying DELMAX's, and Table 2.3 shows the probability distributions assumed for the stochastic variables in this problem.

TABLE 2.2: REGULATORY LEVELS AS A FUNCTION OF DELMAX

Compound	DELMAX(mg/day)	Regulatory Level(ppm)
Compound A	.000001	.014
Compound B	.00001	.14
Compound C	.0001	1.4
Compound D	.001	14
Compound E	.01	140
Compound F	.1	1400
Compound G	1.0	14000

TABLE 2.2

DISTRIBUTIONS OF THE INPUT STOCHASTIC VARIABLES FOR THE REGULATORY PROBLEM

Variable	Type	Outcome	Cumulative Probability
RLEACH	Continuous Uniform	0.5	0.00
		1.0	1.00

Variable	Type	Parameters
PRECIP	Trapazoidal	762.0 (= lowest value) 1016.0 (= most likely value, lower boundary) 1270.0 (= most likely value, upper boundary) 1016.0 (= highest value)
DEPTHL	Trapazoidal	3.0 (= lowest value) 6.1 (= most likely value, lower boundary) 9.1 (= most likely value, upper boundary) 15.2 (= highest value)
WDENS	Triangular	297.0 (= lowest value) 415.0 (= most likely value) 534.0 (= highest value)

2.6 The MOUSE Regulatory Program And Results

Figure 2.5 shows the MOUSE program required for the Regulatory problem. There are 18 lines of computer code but only the following line was written by the user directly (TRAP - see Chapter 11 - with user input, wrote the remainder of the program):

```
FACTOR=70000*(PRECIP*RLEACH)/(DEPTHL*WDENS)
```

A full explanation for the MOUSE program of Figure 2.5 will be given in the Chapters that follow. The output for this program is shown in Figure 2.6. Note that the mean value of FACTOR is estimated as 20361. Since 5000 iterations were used, this is within a fraction of a percent of the true value. The "best" value approach (using the mean values for the four input distributions) produces an average value equal to 17303, i.e., approximately 15% smaller than the true mean. This is due to the non-linear functional relationships reflected in the uncertainty equations.

```
C
C REGULATORY PROBLEM
C
      DIMENSION TAB(46)
      COMMON/Q1/C80/Q2/JUMP/Q3/LOOP/Q4/SENS/Q5/IDATA/Q6/IOUT/Q7/IPASS/Q8
     &/IRAN/Q9/IREAD/Q10/ISCR1/Q11/ISCR2/Q12/ISNAP/Q13/ITER/Q14/ITOG/Q15
     &/IWRIT/Q16/IXXX/Q17/OK1/Q18/OK2/Q19/OK3/Q20/OK4/Q21/OK5/Q22/OK6
      CALL NAME('(REGULATORY PROBLEM)')
      ITER=5000
5     DO 10 IXXX=1,ITER
      PRECIP=TRA(762.0,1016.0,1270.0,1524.0)
      RLEACH=CUNI(.5,1.0)
      DEPTHL=TRA(3.0,6.1,9.1,15.2)
      WDENS=TRI(297.0,415.0,534.0)
      FACTOR=70000*(PRECIP*RLEACH)/(DEPTHL*WDENS)
      CALL TABLE(FACTOR,'(FACTOR)',DUM0,DUM1,40,TAB)
10    CONTINUE
      GOTO(5,15), JUMP
15    CONTINUE
      END
```

FIGURE 2.5

MOUSE PROGRAM FOR THE REGULATORY PROBLEM

```
                     *******************************************
                     *    DISTRIBUTION FOR QUANTITY FACTOR     *
                     *******************************************


NUMBER OF ITERATIONS =  5000          MEAN =     20361.21000             STANDARD DEVIATION =     12154.16000
                                   MINIMUM =      4219.64600       COEFFICIENT OF VARIATION, % =        59.69275
                                   MAXIMUM =     82200.32000

       LOWER NUMBER OF   PERCENT CUMULATIVE CUMULATIVE             DISTRIBUTIONS
       LIMIT   ENTRIES   ENTRIES  % ENTRIES COMPLEMENT   * = FREQUENCY DISTRIBUTION      O = CUMULATIVE DISTRIBUTION

   4200.0000      60.      1.20      1.20     98.80   *O*****
   6100.0000     245.      4.90      6.10     93.90   ****O**********************
   8000.0000     443.      8.86     14.96     85.04   *********O*************************************
   9900.0000     585.     11.70     26.66     73.34   ****************O********************************************
  11800.0000     486.      9.72     36.38     63.62   **********************O*****************************
  13700.0000     467.      9.34     45.72     54.28   ****************************O*********************
  15600.0000     381.      7.62     53.34     46.66   *********************************O*******
  17500.0000     374.      7.48     60.82     39.18   *************************************O**
  19400.0000     248.      4.96     65.78     34.22   ***************************             O
  21300.0000     239.      4.78     70.56     29.44   **************************                 O
  23200.0000     194.      3.88     74.44     25.56   *********************                        O
  25100.0000     158.      3.16     77.60     22.40   *****************                              O
  27000.0000     127.      2.54     80.14     19.86   **************                                   O
  28900.0000     121.      2.42     82.56     17.44   **************                                    O
  30800.0000     107.      2.14     84.70     15.30   ************                                        O
  32700.0000      94.      1.88     86.58     13.42   ***********                                          O
  34600.0000      95.      1.90     88.48     11.52   ***********                                           O
  36500.0000      69.      1.38     89.86     10.14   ********                                               O
  38400.0000      75.      1.50     91.36      8.64   *********                                               O
  40300.0000      70.      1.40     92.76      7.24   ********                                                 O
  42200.0000      56.      1.12     93.88      6.12   *******                                                  O
  44100.0000      56.      1.12     95.00      5.00   *******                                                   O
  46000.0000      55.      1.10     96.10      3.90   *******                                                    O
  47900.0000      35.       .70     96.80      3.20   *****                                                      O
  49800.0000      26.       .52     97.32      2.68   ****                                                       O
  51700.0000      24.       .48     97.80      2.20   ****                                                        O
  53600.0000      28.       .56     98.36      1.64   ****                                                        O
  55500.0000      17.       .34     98.70      1.30   ***                                                         O
  57400.0000      11.       .22     98.92      1.08   **                                                          O
  59300.0000      15.       .30     99.22       .78   ***
  61200.0000       5.       .10     99.32       .68   **
  63100.0000       6.       .12     99.44       .56   **
  65000.0000       9.       .18     99.62       .38   **
    OVERFLOW      19.       .38    100.00       .00   ***


CUMULATIVE  CUMULATIVE         VALUE OF
 % ENTRIES  COMPLEMENT         FACTOR


      5.0        95.0        5673.4690
     10.0        90.0        6936.3430
     25.0        75.0        9630.4280
     50.0        50.0       14767.1900
     75.0        25.0       23536.7100
     90.0        10.0       36677.3300
     95.0         5.0       44100.0000
     99.0         1.0       57906.6700
```

FIGURE 2.6: STATISTICS, HISTOGRAM AND GRAPHS PRODUCED BY MOUSE FOR THE REGULATORY PROBLEM

CHAPTER 3

THE MOUSE LANGUAGE

3.1 Overview of MOUSE Programs

Although MOUSE is written and based upon FORTRAN (and therefore requires a FORTRAN compiler as part of its system), it is designed to be implemented by those with little or no knowledge of computer languages or programming. MOUSE programs consist of:

(a) Eight Control lines (see Figure 3.1).
(b) A Name line, and
(c) Four Program Blocks,

The four blocks are:

(1) Dimension Block,
(2) Specification Block,
(3) Read Block, and
(4) Program Block.

Because we shall have need to refer to actual programs in order to locate these elements of a MOUSE model, the two example problems of Chapter 2 are redisplayed in Figures 3.2 and 3.3 along with Comment lines for clarity. Comment lines are those that start with a "C" in the first column of a program line.The computer ignores such lines but does print them if the program is listed on a printer or on a terminal. You can (and should) use as many comment lines as will help in following or clarifying a given program.

The Dimension Block consists of all lines occurring before Mouse Control line # 1 and contains lines that size or dimension the various tables, vectors, and matrices used by a program for storing data and results. The lines in the Dimension Block are defined by a set of specific rules, but you need not be concerned with their exact meaning. Examples of the lines in the Dimension Blocks of Figures 3.2 and 3.3 include those starting with the label DIMENSION.

The Specification Block informs the computer of certain decisions under your control, such as the number of iterations in the Monte Carlo method, the number of intervals in the table histograms, and the number of columns in the MOUSE output. Examples of the lines in the Specification Blocks of Figures 3.2 and 3.3 include those starting with the label ITER.

```
           *******************
           * DIMENSION BLOCK *
           *******************
                    |
                    |
                   \|/
     MOUSE CONTROL LINES 1, 2, AND 3
                    |
                    |
                   \|/
                NAME LINE
                    |
                    |
                   \|/
        +++++++++++++++++++++++++
        + READ BLOCK (Optional) +
        +++++++++++++++++++++++++
                    |
                    |
                   \|/
         ***********************
         * SPECIFICATION BLOCK *
         ***********************
                    |
                    |
                   \|/
          MOUSE CONTROL LINE 4
                    |
                    |
                   \|/
*****************************************
*             PROGRAM BLOCK             *
* 1. Random Sampling of Input Variables *
* 2. Uncertainty Equations (Model)      *
* 3. Output Requests                    *
* 4. Optional Control lines             *
*****************************************
                    |
                    |
                   \|/
   MOUSE CONTROL LINES 5, 6, 7, and 8
```

FIGURE 3.1

GENERAL OVERVIEW OF A MOUSE PROGRAM

```
C-----------------<Start of DIMENSION BLOCK>--------------------
C
C EMPIRICAL VARIABLE DIMENSION LINE
C
      DIMENSION D1(3,2),D2(4,2),D3(4,2)
C
C TABLE DIMENSION LINE
C
      DIMENSION TAB(46)
C
C-------------------<End of DIMENSION BLOCK>--------------------
C
C MOUSE CONTROL LINES 1, 2, AND 3
C
      COMMON/Q1/C80/Q2/JUMP/Q3/LOOP/Q4/SENS/Q5/IDATA/Q6/IOUT/Q7/IPASS/Q8
     &/IRAN/Q9/IREAD/Q10/ISCR1/Q11/ISCR2/Q12/ISNAP/Q13/ITER/Q14/ITOG/Q15
     &/IWRIT/Q16/IXXX/Q17/OK1/Q18/OK2/Q19/OK3/Q20/OK4/Q21/OK5/Q22/OK6
C
C NAME LINE
C
      CALL NAME('(WASTE OIL RECLAMATION PROBLEM)')
C
C-------------------<Start of READ BLOCK>-----------------------
C
C EMPIRICAL VARIABLE READ LINES
C
      CALL READ(D1,3)
      CALL READ(D2,4)
      CALL READ(D3,4)
C
C--------<End of READ BLOCK & Start of SPECIFICATION BLOCK>------
C
C ITERATION CONTROL LINE
C
      ITER=5000
C
C--------------------<End of SPECIFICATION BLOCK>-----------------
C
C MOUSE CONTROL LINE 4
C
5     DO 10 IXXX=1,ITER
C
C---------------------<Start of PROGRAM BLOCK>------------------
C
C.......<PROGRAM BLOCK: Random sampling of input variables>......
C
C STOCHASTIC VARIABLE SAMPLING LINES
C
      CONTAM=DUNI(1.0,4.0)
      RAND1=CUNI(-10.0,10.0)
      PRICE1=EDIS(D1,3)
      SLOPE=ECTS(D2,4)
      PROD2=STEP(D3,4)
      RAND2=GAU(0.00,0.10)
      PRICE3=EBETA(6.5,8.0,9.5)
      PROD3=EXPON(18.0)
      XLIFE=DTRI(5.0,10.0,15.0)
      XININV=TRA(1600.0,2000.0,2200.0,2400.0)
C
C......... <PROGRAM BLOCK: Uncertainty equations (model)>........
C
      THRU1=10.0-CONTAM
      YIELD1=10.0-2.0*CONTAM
      PROD1=THRU1*YIELD1+RAND1
      GREV1=PRICE1*PROD1
      PRICE2=10.0-SLOPE*PROD2+RAND2
      GREV2=PRICE2*PROD2
      GREV3=PRICE3*PROD3
      ANETB=GREV1+GREV2+GREV3-70.0
      PVALUE=ANETB*UPW(0.08,XLIFE,0.0)-XININV
C
C..............<PROGRAM BLOCK: Output requests>..................
C
C TABLE OUTPUT LINE
C
      CALL TABLE(PVALUE,'(PRES VAL)',DUM0,DUM1,40,TAB)
C
C------------------<End of PROGRAM BLOCK>-----------------------
C
C MOUSE CONTROL LINES 5, 6, 7, AND 8
C
10    CONTINUE
      GOTO(5,15), JUMP
15    CONTINUE
      END
```

FIGURE 3.2:

COMMENTED MOUSE PROGRAM FOR

THE WASTE OIL RECLAMATION PROBLEM

```
C-----------------<Start of DIMENSION BLOCK>--------------------
C
C TABLE DIMENSION LINE
C
      DIMENSION TAB(46)
C
C-------------------<End of DIMENSION BLOCK>--------------------
C
C MOUSE CONTROL LINES 1, 2, AND 3
C
      COMMON/Q1/C80/Q2/JUMP/Q3/LOOP/Q4/SENS/Q5/IDATA/Q6/IOUT/Q7/IPASS/Q8
     &/IRAN/Q9/IREAD/Q10/ISCR1/Q11/ISCR2/Q12/ISNAP/Q13/ITER/Q14/ITOG/Q15
     &/IWRIT/Q16/IXXX/Q17/OK1/Q18/OK2/Q19/OK3/Q20/OK4/Q21/OK5/Q22/OK6
C
C NAME LINE
C
      CALL NAME('(REGULATORY PROBLEM)')
C
C--------<Start of SPECIFICATION BLOCK>------
C
C ITERATION CONTROL LINE
C
      ITER=5000
C
C--------------------<End of SPECIFICATION BLOCK>----------------
C
C MOUSE CONTROL LINE 4
C
5     DO 10 IXXX=1,ITER
C---------------------<Start of PROGRAM BLOCK>------------------
C
C.......<PROGRAM BLOCK: Random sampling of input variables>......
C
C STOCHASTIC VARIABLE SAMPLING LINES
C
      PRECIP=TRA(762.0,1016.0,1270.0,1524.0)
      RLEACH=CUNI(.5,1.0)
      DEPTHL=TRA(3.0,6.1,9.1,15.2)
      WDENS=TRI(297.0,415.0,534.0)
C
C..........<PROGRAM BLOCK: Uncertainty equation (model)>.........
C
      FACTOR=70000*(PRECIP*RLEACH)/(DEPTHL*WDENS)
C
C..............<PROGRAM BLOCK: Output requests>..................
C
      CALL TABLE(FACTOR,'(TEST)',DUM0,DUM1,40,TAB)
C
C------------------<End of PROGRAM BLOCK>-----------------------
C
C MOUSE CONTROL LINES 5, 6, 7, AND 8
C
15    CONTINUE
      GOTO(5,15), JUMP
10    CONTINUE
      END
```

FIGURE 3.3: COMMENTED MOUSE PROGRAM FOR THE REGULATORY PROBLEM

The Read Block enters input data, such as empirical probability distributions, and parameters (i.e., temporary constants) of the model. Since not all MOUSE programs involve empirical probability distributions or parameters, the Read Block may not be needed in all programs; hence the use of the term "optional" in Figure 3.1. The Regulatory problem, for example, does not use a Read Block. Examples of the lines in the Read Block of Figure 3.2 include those starting with the labels CALL READ.

The Program Block is the heart of any MOUSE program and contains four sections, of which the first three are vital:

(1) A section dealing with the random sampling of the input variables,
(2) Your mathematical model (i.e., the set of uncertainty equations),
(3) Your requests for specific kinds of outputs from the MOUSE program, and
(4) Optional Control lines.

The first two sections are clearly identified in Figures 3.2 and 3.3. Example of lines dealing with output requests in the Program Blocks of Figures 3.2 and 3.3 are the ones starting with the label CALL TABLE. In more complex MOUSE problems, there may be (and hence the use of the term "optional" in Figure 3.1) a requirement for additional, specialized control lines such as a "Pass Control" line. No such lines are shown in Figures 3.2 or 3.3; these will be introduced in later chapters. It should be noted that only the Program Block need be prepared by you; the TRAP utility (Chapter 11) will take care of the rest. Actually, TRAP will also prepare part of the Program Block, i.e., Output Requests and Optional Control lines. Therefore, what appears to be complicated coding is, in reality, quite simple and straightforward.

3.2 Arithmetic

Constants in MOUSE are merely numbers; variables, on the other hand, may be thought of as storage locations in the computer. Variable names are denoted by one to six alphanumeric characters (i.e., any combination of letters and numbers, starting with a letter). Both constants and variables may be either integer or decimal. An integer constant is any number without a decimal point; an integer variable is any whose name starts with the letters I, J, K, L, M, or N. This is an important distinction since all arithmetic in MOUSE is done in decimal numbers. This is why XLIFE and XININV were used instead of LIFE and ININV in the coding of the Waste Oil Reclamation problem and why decimal points were used in the constants in the uncertainty equations. On the other hand, certain control functions require the use of integer constants or variables. A summary of the rules for the use of decimal points in MOUSE appears in Table 3.1.

TABLE 3.1 RULES FOR USE OF DECIMAL POINTS IN MOUSE

Decimal Points Used	Decimal Points Not Used
Classical Probability Functions	Column Output line
Correlation Function (with exception of dimension argument)	Dimension Arguments (all)
Data lines (all)	Empirical Probability Functions
IF Statement Operators and Numerical Arguments	GOTO Statement Number
Interest Functions	Iteration line
Numbers Used in Uncertainty Equations	Output Destination line
Scaling Factors in Table and Plot Output Lines	Pass Control line
Sensitivity lines	Plot Output lines (with exception of scaling factor)
Table Output Lines (with exception of number of histogram intervals argument)	Read lines (all)
Trace lines	Seed Reset lines
	Sensitivity Curtailment line
	Snap line

Under certain circumstances, it is necessary to change decimal variables to integer variables and vice versa. This is done by simple renaming, e.g.,

```
XLIFE=LIFE

LIFE=XLIFE
```

In the first case a new variable, the decimal variable XLIFE, is produced from the integer variable, LIFE (without changing LIFE); in the second case, the reverse occurs. To repeat, neither variable on the right-hand side of the equal sign is altered; instead, a new variable is created.

The valid arithmetic operations available in MOUSE are:

Operation	Symbol	Example
Addition	+	A+B
Subtraction	-	A-B
Multiplication	*	A*B
Division	/	A/B
Exponentiation	**	A**B (= A^B)

Remark: MOUSE does not allow two arithmetic operations to appear next to each other. Thus, A*-B must be written as A*(-B).

The conventional use of parentheses in arithmetic expressions is acceptable in MOUSE and, indeed, the use of parentheses is encouraged whenever it contributes to the clarity of an expression. There is a hierarchal sequence of arithmetic operations in MOUSE. Exponentiation is done first, then multiplication or division, then addition or subtraction. Parentheses override this hierarchal sequence, however.

Along with these arithmetic operations, there are also a number of standard functions available as follows:

Function	Definition	Function Name With Argument
square root	$(a)^{0.5}$	SQRT(A)
exponential	e^a	EXP(A)
natural logarithm	ln(a)	ALOG(A)
common logarithm	log(a)	ALOG10(A)
absolute value	$\|a\|$	ABS(A)
sine	sin(a)	SIN(A)
cosine	cos(a)	COS(A)
tangent	tan(a)	TAN(A)

There are other functions available but they are not likely to be encountered in uncertainty equations. The reader is referred to a FORTRAN handbook should such a need arise (see Lipschutz & Poe, 1978).

As an example, consider how the following uncertainty equation would be written in MOUSE:

$$Y_4 = (3X_3)^{0.5} + X_2^{0.8}$$

In MOUSE this equation would appear as

```
Y4 = SQRT(3.0*X3) + X2**0.8
```

Note that there is a multiplication sign between the 3 and the X3, and that a decimal point (and an optional zero for clarity) has been added to the 3 in accordance with the decimal arithmetic rule stated earlier. Note also that the left-hand side of any equation must be a single variable name. One cannot write, for example,

```
4.0*X5 = Y8
```

In general, program lines start in or after column 7 and no entry is permitted after column 72. (Data lines -- see Chapter 4 -- follow a different protocol, however. Statement numbers used in conjunction with GOTO statements are also exceptions. See Chapter 4, Sections 4.6 and 4.7, and Table C.1, Appendix C.) If a particularly long equation is encountered, it can be continued on the next line but the continuation line (or lines) must have a some character (the author recommends a "&") placed in column 6. As many continuation lines as are necessary generally may be used; there are limits, of course, that depend upon the FORTRAN compiler being used but as these limits are fairly generous this usually poses no serious problems. Your local system FORTRAN manuals should be consulted for details. It should also be noted that spaces (blanks) between arithmetic operations are ignored by the computer, e.g., the following are identical:

```
Y=1.0+X3

Y = 1.0 + X3
```

3.3 Restricted Names

The following are <u>not</u> allowed as variable names in your uncertainty equations because they are reserved for special MOUSE program codes:

Q1
Q2
Q3
Q4
Q5
Q6
Q7
Q8
Q9
Q10
Q11
Q12
Q13
Q14
Q15
Q16
Q17
Q18
Q19
Q20
Q21
Q22
C80
JUMP
LOOP
RAN
SENS
IDATA
IOUT
IPASS
IRAN
IREAD
ISCR1
ISCR2
ISNAP
ITER
ITOG
IWRIT
IXXX
OK1
OK2
OK3
OK4
OK5
OK6

There is, furthermore, a prohibition on the use of names used by MOUSE as names of functions and subroutines, and names used in the various MOUSE line types. Alphabetized lists of these names can be found in the tables in Appendix B found at the end of this book.

3.4 Stochastic Input Variables

The method by which stochastic input variables are usually introduced into MOUSE uncertainty equations is indirect, i.e., the probability distributions are first sampled and then the random values produced are assigned to a surrogate variable. Referring to Figure 3.2, although probability distributions will be explained in detail in Chapter 4, it is sufficient to say at this point that DUNI(1.0,4.0) returns a random value from the probability distribution for the variable CONTAM. (In the Waste Oil Reclamation problem, therefore, it will return a value of 1, 25% of the time; a value of 2, 25% of the time; a value of 3, 25% of the time, and a value of 4, 25% of the time.) In other words, one usually samples first and then assigns the random value produced to a surrogate variable, such as CONTAM.

The indirect method is used for two reasons: (a) it minimizes coding errors, and (b) it is the method that is usually the most appropriate for any stochastic input variable that appears more than once in the set of uncertainty equations. For the Waste Oil Reclamation problem, this includes the variables CONTAM and PROD2 since they appear twice in the equations for THRU1 and YIELD1, and PRICE2 and GRREV2, respectively. In model sampling, for any iteration the set of values for the stochastic variables is usually assumed to be constant. It would not make sense, for example, to use one value of CONTAM to calculate the acres of crop 1 and yet another to calculate yield per acre of crop 1. Each time the computer refers to DUNI(1.0,4.0), however, a new value of CONTAM is produced. The indirect method results in only one value being produced since DUNI(1.0,4.0) is used only once per iteration. However, under certain circumstances (see Chapter 10, Section 10.5), the direct method may be safely used.

EXERCISES

Which of the following are valid MOUSE equations?

1. `3RDSHF=HRS**2-SQRT(Y4)`
2. `DEC=TIME*-53.2`
3. `Y=EXP(-COUNT)+ALOG(X26)`
4. `X/2.=A6.*PRICE+SEND`
5. `Q1=RATE+5.*TIME`
6. `COST=BASE+SHIP-DISC`
7. `CONTAM=3.*SQRT(X6)-ABS(X7)`
8. `DISCOUNT=ALOG((X1-X2)/(Y1-Y2)`
9. `SALES=ALOG(Y1+Y2)*PRICE-25.4`

REFERENCE

Lipschutz, S., and Poe, A., Programming With Fortran, Schaum's Outline Series in Computers, McGraw-Hill Book Co., New York, p. 1-314 (1978).

CHAPTER 4

MOUSE LIBRARY PROBABILITY FUNCTIONS

4.1 Introduction

In this chapter, the methods used in MOUSE to obtain random samples from various probability distributions are described. (Techniques to determine which distributions are appropriate are discussed in Chapters 12 and 13.) A probability distribution is any rule which assigns a probability to each possible value of a random variable. Discrete random variables in MOUSE are those that are allowed to take on only selected values, while continuous random variables can take on the value of any real number. MOUSE provides nineteen probability functions that may be called by you when desired. Two empirical and fifteen classical distributions are provided, with two of the classical distributions appearing in alternative forms. Furthermore, any of these probability distributions can be truncated and/or composited. (In addition, two correlated probability functions are available; see Chapter 9.) The general form of a probability function in MOUSE is:

function(argument1,argument2, etc.)

The number of arguments varies from one to four, according to the distribution required. The empirical distributions involve the most input from you and will be discussed first.

4.2 Empirical Probability Functions

A continuous empirical probability random variable is called by:

ECTS(name,n)

and a discrete empirical probability random variable is called by:

EDIS(name,n)

where "name" is the name of a cumulative probability distribution array of size n x 2, and "n" is the number of points in the distribution. Note that the n in both the ECTS and EDIS functions are of integer form, i.e., no decimal points are used. Also note that the name must be a decimal name. (A decimal name is one that starts with any letter except I, J, K, L, M, or N. See Chapter 3, Section 3.2).

As an example, in the waste Oil Reclamation problem a random value of the variable PRICE1 was called by:

```
PRICE1=EDIS(D1,3)
```

In this example, we assigned the name D1 to the cumulative probability distribution for PRICE1, the distribution having 3 observations.

Whenever an empirical probability function is part of a MOUSE program, the cumulative probability function must be dimensioned in a <u>Empirical Variable Dimension</u> line and a corresponding <u>Empirical Variable Read</u> line must be supplied. The Empirical Variable Dimension line has the following form:

```
DIMENSION name1(n1,2),name2(n2,2),name3(n3,2), etc.
```

where "namei" is the name of the ith cumulative probability distribution, and "ni" is the number of points in the distribution. These two arguments must correspond to those used in the calling function. An example is:

```
      DIMENSION D1(3,2),D2(3,2),D3(3,2),D4(3,2),D5(3,2),D6(3,2),
     &D7(3,2),D8(3,2),D9(3,2),D10(3,2)
```

This example dimensions ten probability distributions. The D of DIMENSION starts in column 7; note that an & has been placed in column 6 of the second line since this is a continuation line. Note: See Chapter 11, Section 11.6, for a discussion of TRAP, a MOUSE utility that greatly facilitates the preparation of the Dimension Block.

The <u>Empirical Variable Read</u> line (one is needed for <u>each</u> probability distribution read in) has the following form:

```
CALL READ(name,n)
```

where "name" and "n" have the same definitions as described for the Empirical Variable Dimension line. An example from Figure 3.1 is:

```
CALL READ(D1,3)
```

Empirical Variable Read lines, if there are any in the program, must be placed immediately after the Name Line (see Chapter 7, Section 7.9).

The data for empirical probability distributions are entered, in a free-form format, into <u>Empirical Variable Data</u> lines which form the MOUSE Data File, MOUSE.DAT. Figure 4.1 shows these lines for the Waste Oil Reclamation problem. PRICE1 in this figure is distributed according to a Discrete Empirical probability distribution, and SLOPE is distributed according to the Continuous Empirical probability distribution. (PROD2 is distributed according to the Step Rectangular distribution, which will be discussed shortly.) The value of the variable is entered, starting in column 1, and is followed by a comma and the probability data in its cumulative form. Decimal points are used in <u>all</u> numbers, including zeros. Each probability distribution is preceded by a name line, i.e., a line with information to identify the probability distribution when it is read out via the echo print feature of MOUSE. The name line, which is required (it is <u>not</u> optional), is also shown in Figure 4.1. Data in the name line can start in any column.

It is important that the first Empirical Variable Data line contains the left-most observation of the probability distribution when using the continuous empirical probability function, ECTS. The cumulative probability in this line must perforce be zero since there are no values to the left of this point. For example, if we wish to consider the PRICE1 variable as continuous over the interval 7 to 10.5 rather than discrete for the values

```
PRICE1         (PRICE OF MOTOR OIL GRADE 1)
7.5,0.35
8.0,0.75
10.5,1.00
SLOPE          (SLOPE OF MOTOR OIL GRADE 2 PRICE EQUATION)
0.05,0.00
0.10,0.30
0.15,0.70
0.20,1.00
PROD2          (PRODUCTION OF MOTOR OIL GRADE 2)
30.0,0.00
35.0,0.15
60.0,0.65
65.0,1.00
```

FIGURE 4.1

THE MOUSE.DAT DATA FILE CONTAINING THE EMPIRICAL VARIABLE DATA LINES FOR THE WASTE OIL RECLAMATION PROBLEM

7.5, 8, and 10.5, then the Empirical Variable Data lines for PRICE1 would appear as:

```
PRICE1        (PRICE OF MOTOR OIL GRADE 1)
7.0,0.00
7.5,0.35
8.0,0.75
10.5,1.00
```

and the appropriate Empirical Variable Dimension and Read lines would have to be changed to include arguments (4,2) and (D1,4), respectively. TRAP (see Chapter 11) can be used to prepare MOUSE data files. This program not only performs the probability cumulation for you, but also inserts the zero probability point in continuous empirical probability functions in its proper place in the data set.

4.3 A Semi-Empirical Probability Function: The Step Rectangular Function

The step rectangular distribution is a combination of empirical and classical probability functions (the classical function embedded in this distribution is that of the uniform continuous kind), but it is treated in MOUSE more like the empirical distributions just discussed than the classical distributions to be described in Section 4.4. An example of a step rectangular probability distribution is shown in Figure 4.2. In this distribution, discrete probabilities are assigned to ranges of the variable, with the assumption that within each range the variable is distributed uniformly.

The step rectangular probability function is called by:

```
STEP(name,n)
```

where "name" and "n" have definitions identical to those given in Section 4.2. The data for step rectangular distributions are handled exactly as are those for continuous empirical probability functions, i.e., the distribution must be dimensioned in an Empirical Variable Dimension line, an Empirical Variable Read line must be used, and the data must be entered via Empirical Variable Data lines. For the diagram above, these first two lines might appear (in the appropriate places in the program) as:

```
DIMENSION D1(4,2)
      .
      .
CALL READ(D1,4)
      .
      .
```

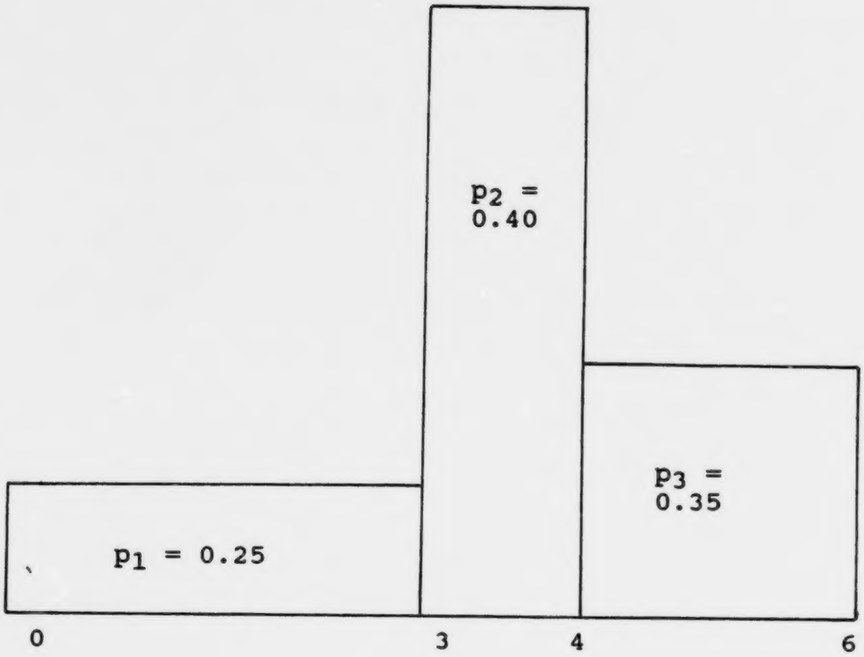


FIGURE 4.2:

EXAMPLE OF A STEP RECTANGULAR PROBABILITY DISTRIBUTION

The Empirical Variable Read lines would appear as follows:

```
X1          STEP RECTANGULAR DISTRIBUTION
0.0,0.00
3.0,0.25
4.0,0.65
6.0,1.00
```

As with continuous empirical probability distribution data lines, the first cumulative probability must be set equal to zero. (TRAP - see Chapter 11 - can do the probability cumulation for you, and also insert the zero probability point in the step rectangular probability function in its proper place in the data set.)

4.4 Classical Probability Functions: Continuous

Classical probability functions (discrete or continuous) supplied by MOUSE do not require Dimension, Read or Data lines. You supply the parameters of these distributions in the arguments of the calling functions directly via calling arguments. Arguments to classical probability distribution calls, however, can be simple or complex. For example, a simple call to a normal probability distribution with mean 50 and standard deviation 10 might look like the following:

```
VAR=GAU(50.0,10.0).
```

However, the arguments need not be direct; they can be functions of other variables, e.g.,

```
VAR=GAU(PRICE/5.0,30.0*COST).
```

In any case, decimal points are used in all numbers that appear in the arguments.

4.4.1 Normal Distribution:

The normal distribution is called by:

```
GAU(mean,std)
```

where "mean" is the mean of the distribution and "std" is its standard deviation.

4.4.2 Exponential Distribution:

The exponential distribution is called by:

```
EXPON(mean)
```

4.4.3 Continuous Uniform Distribution:

The continuous uniform distribution is called by:

```
CUNI(lower,upper)
```

where "lower" is the lower bound and "upper" is the upper bound of the distribution. The mean and variance of the continuous uniform distribution are:

$$\text{mean} = (\text{upper} + \text{lower})/2 \qquad [4.1]$$

$$\text{var} = (\text{upper} - \text{lower})^2/12 \qquad [4.2]$$

4.4.4 Lognormal Distribution:

The lognormal distribution is called by:

`GAUL(mean,std)`

4.4.5 Lognormal Distribution, Alternate form:

An alternate form of the lognormal distribution is called by:

`AGAUL(mode, median)`

where "mode" and "median" are the mode and median, respectively, of the distribution. Note that the median must be greater than the mode or else an error message will occur and the MOUSE program will be terminated automatically.

4.4.6 Beta Distribution, Standard Form:

The general mathematical form of the beta distribution is:

$$f(x) = \frac{x^{a-1}(1-x)^{b-1}}{B(a,b)} \qquad [4.3]$$

where a and b are parameters greater than zero and where B(a,b) is the beta function. The standard form of the beta distribution is called by:

`BETA(a,b)`

The mean and variance of the standard beta distribution are:

$$\text{mean} = a/(a + b) \qquad [4.4]$$
$$\text{var} = ab(a + b)^{-2}(a + b - 1)^{-1} \qquad [4.5]$$

4.4.7 Beta Distribution, Empirical Form:

The empirical form of the beta distribution is called by:

`EBETA(l,m,u)`

where l is the lowest value the variable can reasonably assume, m is the most likely value, and u is the highest value the variable can reasonably assume. MOUSE will automatically scale and fit an appropriate beta distribution when presented with these three parameters. The method works, however, only when the parameters of the underlying standard beta distribution both are greater than zero. If MOUSE finds that either of these parameters is less than zero, it will so inform you and terminate the run. The mean and standard deviation of the empirical beta distribution are:

$$\text{mean} = (4m + l + u)/6 \qquad [4.6]$$
$$\text{std} = (u - l)/6 \qquad [4.7]$$

4.4.8 Gamma Distribution:

The general mathematical form of the gamma distribution is:

$$f(x) = \frac{x^{a-1}e^{-x/b}}{b^{a}G(a)} \qquad [4.8]$$

where a and b are parameters greater than zero and where G(a) is the gamma function. The gamma distribution is called by:

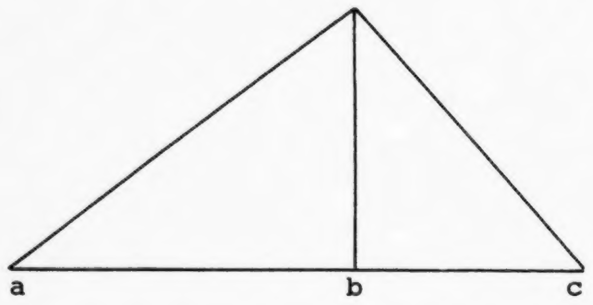
GAM(a,b)

The mean and variance of the gamma distribution are:

$$\text{mean} = ab \qquad [4.9]$$
$$\text{var} = ab^2 \qquad [4.10]$$

4.4.9 Triangular Distribution:

The triangular distribution has the following shape:



The triangular distribution is called by:

TRI(a,b,c)

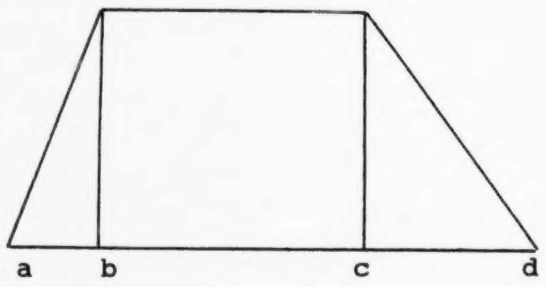
The mean and variance of the triangular distribution are:

$$\text{mean} = (a + b + c)/3 \qquad [4.11]$$

$$\text{var} = [a(a - b) + c(c - a) + b(b - c)]/18 \qquad [4.12]$$

4.4.10 Trapezoidal Distribution:

The trapezoidal distribution has the following shape:



The trapezoidal distribution is called by:

TRA(a,b,c,d)

4.5 Classical Probability Functions: Discrete

The discrete classical probability functions supplied by MOUSE also do not require Dimension lines, Read lines or Data lines. You supply the parameters of these distributions in the arguments of the calling functions directly. Decimal points are used in all discrete classical probability functions.

4.5.1 Erlang Distribution:

The general mathematical form of the erlang distribution is:

$$f(x) = \frac{x^{k-1}e^{-x/a}}{a^k(k-1)!} \qquad [4.13]$$

where a and k are greater than zero and k is an integer (but, as in all MOUSE probability functions, must be written with a decimal point). The erlang distribution is called by:

ERL(a,k)

The mean and variance of the erlang distribution are:

$$\text{mean} = ka \qquad [4.14]$$
$$\text{var} = ka^2 \qquad [4.15]$$

4.5.2 Poisson Distribution:

The Poisson distribution is called by:

POI(mean)

4.5.3 Binomial Distribution:

The binomial distribution is called by:

BIN(p,n)

where n and p are the usual parameters of the binomial distribution (i.e., p is the probability of a success and n is the number of trials). The mean and variance of the binomial distribution are:

$$\text{mean} = np \qquad [4.16]$$
$$\text{var} = np(1 - p) \qquad [4.17]$$

4.5.4 Pascal/Geometric Distribution:

The general mathematical form of the Pascal distribution is:

$$f(x) = \binom{k+x-1}{x} p^k q^x \qquad [4.18]$$

and describes the number of failures (x) that occur before k successes have occurred (where p is the probability of a success). When k = 1, the Pascal distribution becomes the Geometric distribution. The pascal/geometric distribution is called by:

PAS(k,p)

Once again, k and p are greater than zero and k is an integer (written with a decimal point). The mean and variance of the pascal/geometric distribution are:

$$\text{mean} = k(1 - p)/p \qquad [4.19]$$
$$\text{var} = k(1 - p)/p^2 \qquad [4.20]$$

4.5.6 Hypergeometric Distribution:

The hypergeometric distribution is called by:

HYP(N,n,p)

where N is the total population size, n is the sample size, and p is the probability of success. N, n, and p must be greater than zero, and N and n are integers (written with a decimal point).

4.5.7 Discrete Uniform Distribution

The discrete uniform distribution is called by:

DUNI(lower,upper)

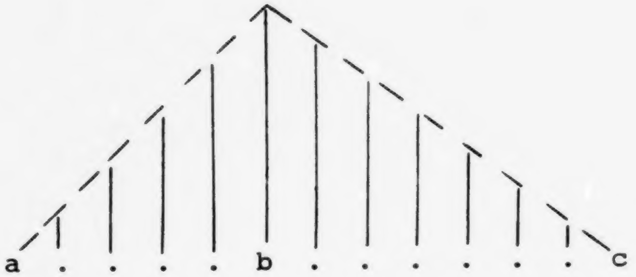
where "lower" is the lower bound and "upper" is the upper bound of the distribution.

4.5.8 Discrete Triangular Distribution

The discrete triangular distribution is called by:

DTRI(a,b,c)

where a, b, and c are the parameters shown as follows:

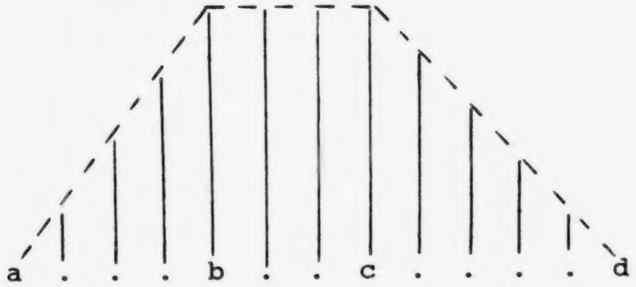


4.5.9 <u>Discrete Trapezoidal Distribution</u>

The discrete trapezoidal distribution is called by:

`DTRA(a,b,c,d)`

where a, b, c, and d are the parameters shown as follows:



4.6 Truncated Distributions

Any distribution can be truncated easily in MOUSE by using an IF statement. The rationale is simply to discard any values generated by the probability function that fall outside of the truncation limits. There are three forms of truncation:

```
IF(variable .LT. lower) GOTO n
IF(variable .GT. upper) GOTO n
IF(variable .LT. lower .OR. variable .GT. upper) GOTO n
```

where "variable" is the name of the random variable to be truncated; "lower" and "upper" are the lower and upper truncation limits, respectively; and "n" is a statement number consisting of an integer between 1 and 9999 (except 5, 10, 15, and 100 which are reserved for certain MOUSE control lines; see Table B.1 in Appendix B for a summary of the use of these reserved statement numbers). The integer n is also placed, starting in column 1, in the line initially generating the random variable. The conditional operators, .LT., .GT., and .OR. (which signify "less than", "greater than", and "or", respectively), are written with the dots or decimal points; the lower and upper limits must contain decimal points, also.

For example, to generate random values of price (PRICE) from a truncated normal distribution with mean 100 and standard deviation 20 such that no values less than 80 will be produced, we write:

```
column 1--       --column 7 (or beyond)
          |     |
         \|/   \|/
          200   PRICE=GAU(100.,20.)
                IF(PRICE .LT. 80.) GOTO 200
```

In other words, if PRICE is less than 80, control of the program reverts to the line with statement number 200, i.e., the price-generation line.

If, in addition, we wanted to generate random values of demand (DEMAND) from an exponential distribution with mean 5 such that no values greater than 10 will be produced, we write:

```
200   PRICE=GAU(100.,80.)
      IF(PRICE .LT. 80.) GOTO 200
201   DEMAND=EXP(5.)
      IF(DEMAND .GT. 10.) GOTO 201
```

4.7 Composite Distributions

Composite distributions are mixtures of two or more distributions, the most commonly encountered being the bimodal type. For example, in many production facilities the quantity produced per unit of time is bimodally distributed with the first peak occurring at quantity equal to zero. This occurs when the production line closes down for repair or preventive maintenance.

For composites consisting of a mixture of just two probability distributions, another form of the IF statement can easily be used to generate the appropriate random values:

```
variable=function1
IF(RAN(IRAN) .LE. w) variable=function2
```

where "function1"and "function2" refers to the two functions involved and "w" is the fraction of the time that function2 is to be used. [RAN(IRAN) is the MOUSE random number call; see Appendix B, Section B.3.] The conditional operator, .LE. (which signifies "less than or equal to"), as well as the w require the decimal point. For example, a 50-50 mixture of normal distributions with means/standard deviations of 100/20 and 200/45, respectively, could be generated by:

```
PRICE=GAU(100.,20.)
IF(RAN(IRAN) .LE. 0.5) PRICE=GAU(200.,45.)
```

In the production example cited above suppose that, when the line was functioning, the quantity produced followed a normal distribution with mean 50 and standard deviation of 15, and that the line was down 10% of the time. The quantity would be generated by:

```
QUANTITY=GAU(50.,15.)
IF(RAN(IRAN) .LE. 0.10) QUANTITY=0.0
```

More complex composites can be generated using this method; the following are the conditional operator codes that are available to you for the construction of such composites:

Operator	Meaning
.EQ.	equal to
.NE.	not equal to
.GT.	greater than
.GE.	greater than or equal to
.LT.	less than
.LE.	less than or equal to
.OR.	or
.AND.	and

The compositing and truncating functions illustrate the use of two powerful instructions in MOUSE: the IF statement and the GOTO statement. The computer normally executes a program line by line, but both the order of execution and the execution itself can be altered by these transfer functions. The GOTO statement illustrated in Section 4.5 imposes an unconditional transfer of control; the IF statement imposes a condition on the operation. In Chapter 10, Section 10.4, an additional use of the IF statement is described. Further uses of the IF and GOTO statements are found in Exercises Nos. 8 and 9 of Chapter 10. The use of IF and GOTO statements can easily transform MOUSE into a more general purpose simulation language.

4.8 Examples of Probability Function Calls

The following stochastic variable sampling lines, along with their appropriate Dimension and Read lines, were taken from the Waste Oil Reclamation Problem shown in Figure 3.2:

```
C
C EMPIRICAL VARIABLE DIMENSION LINES
C
      DIMENSION D1(3,2),D2(4,2),D3(4,2)
C
C EMPIRICAL VARIABLE READ LINES
C
      CALL READ(D1,3)
      CALL READ(D2,4)
      CALL READ(D3,4)
C
C STOCHASTIC VARIABLE SAMPLING LINES
C
      CONTAM=DUNI(1.0,4.0)
      RAND1=CUNI(-10.0,10.0)
      PRICE1=EDIS(D1,3)
      SLOPE=ECTS(D2,4)
      PROD2=STEP(D3,4)
      RAND2=GAU(0.00,0.10)
      PRICE3=EBETA(6.5,8.0,9.5)
      PROD3=EXPON(18.0)
      XLIFE=DTRI(5.0,10.0,15.0)
      XININV=TRA(1600.0,2000.0,2200.0,2400.0)
```

In this case Empirical Variable Dimension and Read lines were necessary since three probability distributions were of the empirical or semi-empirical type. On the other hand, the following probability function calls were taken from the Regulatory Problem shown in Figure 3.3:

```
C
C STOCHASTIC VARIABLE SAMPLING LINES
C
      PRECIP=TRA(762.0,1016.0,1270.0,1524.0)
      RLEACH=CUNI(.5,1.0)
      DEPTHL=TRA(3.0,6.1,9.1,15.2)
      WDENS=TRI(297.0,415.0,534.0)
```

No Stochastic Variable Dimension or Read lines were necessary here because none of the distributions (Trapezoidal, Continuous Uniform, and Triangular) used required them.

4.9 Notes On The Generation Of Stochastic Variates in MOUSE, and Checks On MOUSE Probability Call Syntax

The method of generation of the normal distribution is the modified Box-Mueller algorithm of Marsaglia and Bray (1964); for the gamma distribution the combination method described by Pritsker and Pegden (1979) is used; and for the exponential distribution the method is that of Von Neumann (1951). Those for the empirical beta, discrete uniform, triangular, and trapezoidal were developed by the author (the first is based on re-scaling the standard beta distribution to an arbitrary interval, and the last three are based on the inverse transformation method). The methods used for the other distributions can be found in Naylor, et al. (1966).

A utility program, CHECKER (see Chapter 11, Section 11.2), can be used to examine any MOUSE source program to insure that: (1) the proper number of arguments are made to MOUSE probability calls, (2) the arguments are reasonable (e.g., no negative standard deviations, "upper" is greater than "lower", etc.), the proper form of argument is used, i.e., decimal versus integer number or variable), and (4) the dimensions for empirical probability distributions agree with those specified in the Dimension Block.

4.10 Fitting Classical Probability Distributions With IMP

IMP (Interactive Modeler for Probabilities) is a companion system to MOUSE that serves a variety of functions that deal with classical probability distributions. Unlike MOUSE, the user does not write programs for IMP; it is an interactive system that queries the user for information and data.

The user has a choice between placing the observations into a data file (named IMP.DAT) accessible to the program or entering the data directly from the keyboard. The first line of the data file contains descriptive information (up to 60 characters) used to identify the file. (This line is mandatory.) The remainder of the file contains the sample data, one observation per line starting in column 1, with the use of the decimal point mandatory. Figure 4.3 shows such a file for part of a 100-observation data set consisting of numbers taken at random from a normal distribution with mean = 10 and standard deviation = 5.

IMP prompts the user for instructions and offers a series of menus and options as follows:

1. After calling IMP (simply by typing "IMP"), you will be asked to supply a title or comment line (not to exceed 60 characters) that will be placed in the IMP output file. If the file IMP.OUT (i.e., the standard IMP output file) exists, you will be asked whether you wish to overwrite this file or not. If not, you will asked for the name of an alternate output file.

2. The following menu is presented:

```
**********************************************
* FIT A DATA SET TO A CLASSICAL DISTRIBUTION *
* INTEGRATE A CONTINUOUS  DISTRIBUTION       *
* ANALYZE A BIVARIATE DISTRIBUTION           *
* ANALYZE A DATA SET FOR AUTOCORRELATION     *
**********************************************
  Use up-arrow and down-arrow keys to move bar
  to desired selection.  Then press Enter key.
  The PgUp and PgDn keys will move the bar to
  the top and bottom positions, respectively.
```

   The choices on the menu are highlighted. For the purpose fitting a classical distribution to sample data, select the first option, "FIT A DATA SET TO A CLASSICAL DISTRIBUTION".

2. IMP next asks whether the data are continuous or discrete. In this example the data are continuous since the data set contains 100 random values taken from a normal distribution, a portion of which has been presented in Figure 4.3.

3. You are then queried as to whether you wish to enter the data via the keyboard or via a data file. If you select the former, the data is entered exactly as it would be if a data file were being prepared.

```
FITTING A DATA SET TO A CLASSICAL DISTRIBUTION: NORMAL (100,25)
 89.48
 97.07
106.23
 90.26
107.13
125.30
 83.86
 81.47
112.08
103.82
121.52
104.38
 84.66
108.69
131.95
100.78
123.32
107.21
 87.83
105.28
 81.05
104.12
 86.37
117.23
109.65
 66.40
 77.83
 86.15
 65.72
 82.34
 95.85
```

FIGURE 4.3

PARTIAL DATA FILE FOR FITTING A CLASSICAL DISTRIBUTION USING IMP

4. IMP then asks whether you wish to review or edit the data just entered via keyboard or file. This provides an opportunity to correct erroneous entries and to add or delete a data line. The editor is self-prompting, so no further details will be given here. IMP then prints out the following statistics (taken from the example data set):

```
*******************************************************************
*                     INPUT DATA STATISTICS                       *
*                                                                 *
* NUMBER OF OBSERVATIONS                    =                  50 *
* MEAN                                      =           101.13080 *
* MEDIAN OF INPUT DISTRIBUTION              =           100.66500 *
* STANDARD DEVIATION                        =            21.35185 *
* MINIMUM                                   =            57.36000 *
* MAXIMUM                                   =           151.59000 *
*******************************************************************
```

5. IMP will now ask whether you wish the data plotted. If the data are continuous, it will ask for the number of intervals desired in the histogram; if the data are discrete it will simply ask whether you wish the data plotted or not. In the case of continuous data, the user may have the data re-plotted, using a different number of intervals in the histogram.

6. IMP then presents you with a list of options as follows:

```
********************************************
*                OPTIONS                   *
* TEST FOR CONTINUOUS UNIFORM DISTRIBUTION *
* TEST FOR NORMAL DISTRIBUTION             *
* TEST FOR LOGNORMAL DISTRIBUTION          *
* TEST FOR EXPONENTIAL DISTRIBUTION        *
* TEST FOR EMPIRICAL BETA DISTRIBUTION     *
* TEST FOR GAMMA DISTRIBUTION              *
* TEST FOR ERLANG DISTRIBUTION             *
* TEST FOR ALL OF ABOVE DISTRIBUTIONS      *
* QUIT                                     *
********************************************
```

The options enable to user to:
  a. Quit or exit from IMP;
  b. Test for any of seven continuous distributions (uniform, normal, lognormal, exponential, empirical beta, gamma, and erlang);
  c. Test for all seven continuous distributions.

7. If any of the testing options are selected, IMP will query the user as to whether he wishes a plot of the original data along with the fitted distribution. If plotting is selected, you will be asked to specify the number of intervals desired in the histogram that will be plotted. IMP will then keep the user informed as to the distribution it is fitting if it is not printing other information at the time. Figure 4.4 shows three such plots; a fit of the uniform distribution to the data, a fit of the normal distribution, and a fit of the

```
HISTOGRAM OF ORIGINAL DATA AND FITTED UNIF/CTS DISTRIBUTION.
          (CIRCLES = FITTED, ASTERISKS = ORIGINAL.)

   60.50100      *******            0
   66.78300      ************       0
   73.06500      ************       0
   79.34700      *******************0**********
   85.62900      *******************0***************
   91.91100      *******************0****
   98.19300      *******************0***************
  104.47500      *******************0*********************
  110.75700      *******************0****
  117.03900      ****************** 0
  123.32100      *******************0****
  129.60300      ************       0
  135.88500      *******            0
  142.16700      ************       0
  148.44900      *******            0
```

```
HISTOGRAM OF ORIGINAL DATA AND FITTED NORMAL   DISTRIBUTION.
          (CIRCLES = FITTED, ASTERISKS = ORIGINAL.)

   60.50100      ******0
   66.78300      *********0**
   73.06500      ************  0
   79.34700      ********************0*********
   85.62900      **************************0********
   91.91100      ************************       0
   98.19300      *********************************0*
  104.47500      *********************************0*******
  110.75700      ************************      0
  117.03900      ******************       0
  123.32100      *******************0****
  129.60300      ************  0
  135.88500      *******  0
  142.16700      *****0******
  148.44900      ***0***
```

```
HISTOGRAM OF ORIGINAL DATA AND FITTED BETA     DISTRIBUTION.
          (CIRCLES = FITTED, ASTERISKS = ORIGINAL.)

   60.50100      ******0
   66.78300      ***********0
   73.06500      ************     0
   79.34700      **********************0*******
   85.62900      **************************0********
   91.91100      ************************     0
   98.19300      *******************************0***
  104.47500      ******************************0**********
  110.75700      ************************    0
  117.03900      ******************      0
  123.32100      ********************0***
  129.60300      ************   0
  135.88500      *******    0
  142.16700      *******0****
  148.44900      ***0***
```

FIGURE 4.4: IMP PLOTS OF FITTED DISTRIBUTIONS TO ORIGINAL DATA

empirical beta distribution. The normal distribution is obviously better than the uniform distribution; whether it is better than the beta is not clear at this point.

8. After each distribution (or set of distributions under the ALL option) have been plotted (assuming the plot option was selected), IMP prints out the table shown in Figure 4.5. For each distribution fitted, a Goodness-of-Fit statistic known as the "Foutz Value" is shown as well as the parameters of the fitted distribution. (Goodness-of-Fit statistics are discussed in Section 12.2 of Chapter 12.) The parameters are given in the same order of the probability function arguments given in Chapter 4. For example, for the Uniform Continuous distribution, Parameter 1 is "lower" and Parameter 2 is "upper" as described for that distribution in Chapter 4. For the Normal distribution, Parameters 1 and 2 are "mean" and "std", respectively; for the empirical beta distribution, Parameters 1, 2, and 3 are "l", "m", "u", respectively. The numbers in parentheses under the Foutz value is the significance level of this statistic. (A single asterisk signifies a level between 1% and 5%; a double asterisk signifies a level of 1% or lower.

```
    SUMMARY OF FITS OF CLASSICAL DISTRIBUTIONS TO DATA SET

DISTRIBUTION  FOUTZ VALUE   PARAMETER 1   PARAMETER 2   PARAMETER 3

    UNIF/CTS       3.3865       57.3600      151.5900
                 ( .9999)
    NORMAL          .3071      101.1308       21.3518
                 ( .0388)*
    BETA            .3250       98.8574       41.6219      169.7330
                 ( .1059)
    EXPONEN         .6610      101.1308
                 ( .9999)
    LOGNORM         .3168      101.1308       21.3518
                 ( .0691)
    GAMMA           .3307       22.4334        4.5080
                 ( .1398)
    ERLANG          .3365        4.5080       22.0000
                 ( .1806)
```

FIGURE 4.5:

SAMPLE IMP OUTPUT, CONTINUOUS DISTRIBUTION

The Foutz statistic was chosen for use with continuous distributions because the author has found that it performs better under many circumstances than the more usual Kolmogorov-Smirnov statistic. In Figure 4.5 the minimum Foutz value is that for the normal distribution so that based upon the Foutz value alone, the best fit is that of the normal distribution (although the beta distribution is very close). The numbers in parentheses under the Foutz statistic show the probability of rejecting the null hypothesis (i.e., that there is no difference between the distribution from which the sample data was obtained and the postulated distribution) incorrectly for the Foutz values shown. Since the point to a goodness-of-fit test is not to reject the null hypothesis, these significance levels generally should be set fairly high (say at the .10 level or higher) for decision purposes. based on the Foutz values in Figure 4.5 we most likely would reject all hypotheses except those of the normal and empirical beta distributions. However, goodness-of-fit tests are not all that reliable for small data sets. It is suggested that the fit of a classical distribution to sample data not be made solely upon the basis of a classical test of hypothesis; the user should examine the plots of the original data versus the fitted distribution, the sample statistics calculated from the data, and anything else the user knows or believes about the distribution in question before a final decision is made.

The IMP example of this Section involved a continuous distribution. For discrete distributions only two differences should be noted:

1. IMP does not query the user for the number of intervals to be used when plotting discrete data;
2. IMP test for four discrete distributions (pascal/geometric, poisson, binomial, and uniform);
3. In the Summary Of Fits Of Classical Distributions To Data Set (Figure 4.6), the Chi-Square value is printed out rather than the Foutz value, and the degrees of freedom (d.f.) for the test are also indicated. The principle is the same; that distribution with the smallest Chi-Square value is given the greatest credence. As with continuous data, the numbers in parentheses under the Chi-Square value is the significance level of this statistic. (A single asterisk signifies a level between 1% and 5%; a double asterisk signifies a level of 1% or lower.

Figure 4.6 is an example of IMP output for a discrete data set of 100 numbers, based on the binomial distribution with parameters p=0.3 and n=10:

```
                    INPUT DATA STATISTICS

   NUMBER OF OBSERVATIONS                    =              100
   MEAN                                      =          3.05000
   MEDIAN OF INPUT DISTRIBUTION              =          3.00000
   STANDARD DEVIATION                        =          1.45904
   MINIMUM                                   =           .00000
   MAXIMUM                                   =          6.00000


    SUMMARY OF FITS OF CLASSICAL DISTRIBUTIONS TO DATA SET

 DISTRIBUTION     CHI-SQUARE    PARAMETER 1   PARAMETER 2       d.f.

   GEO/PAS      DISTRIBUTION CANNOT BE FITTED TO THESE DATA.
   POISSON         4.6158        3.0500                       5.0000
                  ( .5355)
   BINOMIAL        2.1074         .3020       10.0000         4.0000
                  ( .2840)
   UNIF/DIS       34.5400         .0000        6.0000         4.0000
                  (1.0000)
```

FIGURE 4.6:

SAMPLE IMP OUTPUT, DISCRETE DISTRIBUTION

The lowest Chi-Square value is for the binomial distribution. Note that the parameters (p=0.3 and n=10) have been estimated very well. Note that the pascal/geometric distribution could not be estimated because the mean of the distribution was 3.05 and the variance was $(1.45)^2=2.12$. For a pascal/geometric distribution, the variance must be greater than the mean.

The following restrictions in using IMP should be noted:

1. The number of histogram intervals that can be processed by IMP is between 2 and 100. If 1 is entered, it will automatically be changed to 2; is a number greater than 100 is entered, it will automatically be changed to 100.

2. The number of histogram intervals must be less than or equal to one-third the number of observations or else an error message will be returned.

3. IMP will not accept fewer than 10 or more than 5000 observations. If fewer are indicated, an error message will be returned, and if more are indicated only the first 5000 will be read.

Finally, before fitting a distribution to a data set, the data should be checked for autocorrelation using the option, "ANALYZE A DATA SET FOR AUTOCORRELATION" (see Chapter 13, Section 13.9).

Note: The binomial, pascal/geometric, beta, poisson, and gamma probability generators used for graphing the fitted distributions in IMP are from Peizer and Pratt (1968); the generator for the normal (and, indirectly, the lognormal) distribution is taken from Burr (1967).

If it desired to subject output data to IMP analysis, data can be collected using an *Accumulation Line*, which has the following form:

```
CALL ACCUM(var1,var2,'(fname)','(info)')
```

where "var1" and "var2" are the names of the variables whose values are to be collected, "fname" is the name of the file which is to contain the data, and "info" is a descriptive information string (up to 60 characters) to be placed in the first line of the file. Note that both "fname" and "info" are literals so the single quotes and parentheses *must* accompany these items *exactly* in the order shown.

The following example of the use of the Accumulation Line is taken from the Waste Oil Reclamation Problem of Figure 2.1 where it is assumed that the variable PVALUE is to be sujected to IMP analysis:

```
      PVALUE=ANETB*UPW(0.08,XLIFE,0.0)-XININV
      CALL ACCUM(PVALUE,0.,'(IMP.DAT)','(CLASSICAL DISTRIBUTION FIT TO T
     &HE VARIABLE PVALUE)')
```

This will place a list of PVALUEs in the file IMP.DAT. Since the second variable is not needed, any value can be used. For convenience, a value of zero was used in the example. This will produce a file (IMP.DAT) will two columns of data, the second consisting of zeros, but when fitting a classicial distrbution, IMP will not read in the second column. Also notice that a continuation line was used (see Chapter 3, Section 3.2) since the Accumulation Line exceeded 72 characters.

4.11 Fitting Empirical Continuous Probability Distributions With IMP

Option 2 in IMP ("INTEGRATE A CONTINUOUS DISTRIBUTION") is of use when it is desired to transfer an empirical continuous distribution from graph paper to MOUSE. With a little practice, some analysts become proficient in drawing the shape of a subjective continuous probability distribution directly on graph paper

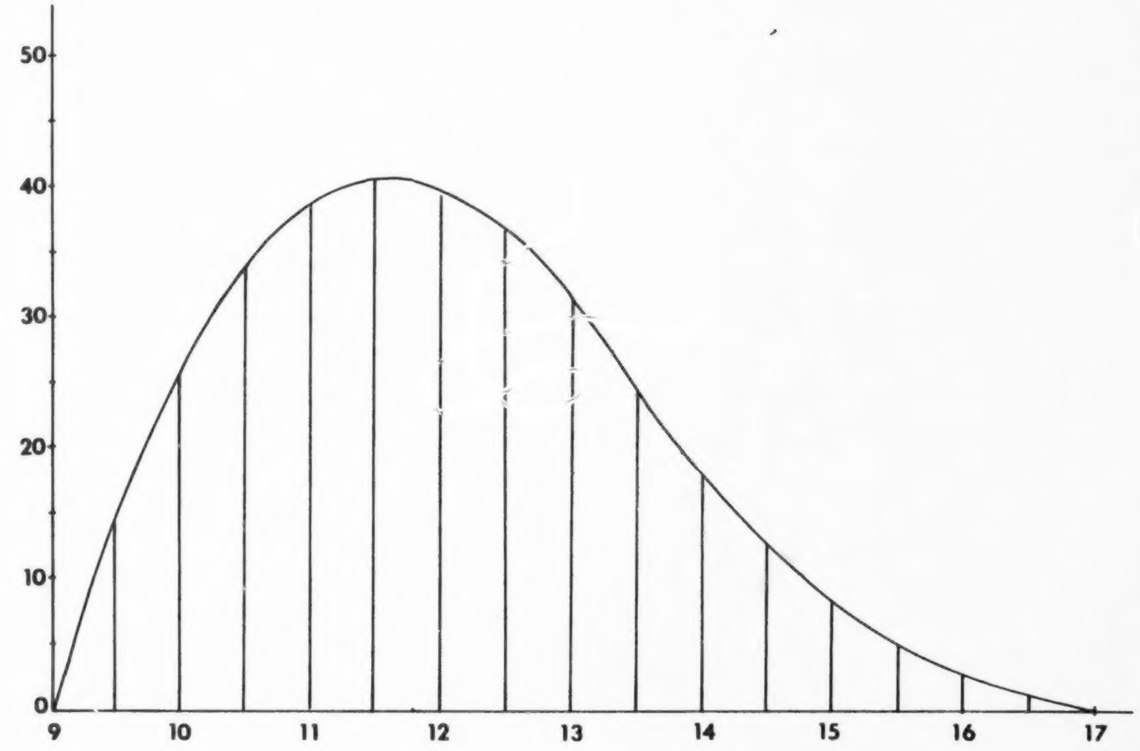


FIGURE 4.7: INTEGRATING AN EMPIRICAL CONTINUOUS PROBABILITY DISTRIBUTION

```
INTEGRATE A CONTINUOUS DISTRIBUTION
9,0
9.5,14
10,26
10.5,34
11,39
11.5,41
12,40
12.5,37
13,32
13.5,24.5
14,17.7
14.5,12.1
15,8
15.5,5
16,2.7
16.5,.8
17,0
```

FIGURE 4.8:

INPUT DATA FOR AN EMPIRICAL CONTINUOUS PROBABILITY DISTRIBUTION

(i.e., the "portrait" approach of Pouliquen referred to in Section 13.6). The human eye is capable of absorbing a great deal of information, and if the subjective curve so drawn truly reflects the analyst's beliefs, then the only remaining task is to transform the distribution to a form usable by MOUSE.

For illustrative purposes, we assume that the analyst has drawn the pseudo-continuous probability distribution shown in Figure 4.7. The term "pseudo" is used since the Y-axis is of arbitrary scale. Under such circumstances the area under the curve will not necessarily add up to 1 and therefore the curve will not be a true probability density function. Scaling can be accomplished by several methods. Since MOUSE requires the cumulative distribution, one method that can be used is to cut the figure into vertical segments and weight them on a sensitive balance. The weight of each individual segment, divided by the total weight, gives the probability of each segment. Subsequently, these probabilities may be cumulated for entry into MOUSE empirical continuous probability distribution data cards. To spare the user from such tedium, option 2 on the IMP Main Menu is used. First, however, the analyst draws a number of <u>equi-spaced</u> vertical lines on his curve (except perhaps for the last interval which, of course, is terminated when the height of the distribution is again zero), as shown in Figure 4.7. (The greater the number of lines drawn, the more accurately the distribution will be represented in MOUSE.) The heights of the lines are read off on the arbitrary Y-axis, and the set of (X,Y) values are placed

```
HISTOGRAM OF INTEGRATED EMPIRICAL CONTINUOUS DISTRIBUTION


   9.00000     *
   9.50000     ********
  10.00000     *********************
  10.50000     *******************************
  11.00000     *************************************
  11.50000     *****************************************
  12.00000     *****************************************
  12.50000     ***************************************
  13.00000     ***********************************
  13.50000     *****************************
  14.00000     **********************
  14.50000     ****************
  15.00000     ***********
  15.50000     *******
  16.00000     *****
  16.50000     ***
  17.00000     *


    ----------------------------------------------------
      INTEGRATED CONTINUOUS EMPIRICAL  DISTRIBUTION
    ----------------------------------------------------

        VALUE OF VARIABLE        CUMULATIVE PROBABILITY

                  9.00000                        .00000
                  9.50000                        .02097
                 10.00000                        .08089
                 10.50000                        .17076
                 11.00000                        .28011
                 11.50000                        .39994
                 12.00000                        .52127
                 12.50000                        .63661
                 13.00000                        .73996
                 13.50000                        .82460
                 14.00000                        .88781
                 14.50000                        .93244
                 15.00000                        .96255
                 15.50000                        .98203
                 16.00000                        .99356
                 16.50000                        .99880
                 17.00000                       1.00000
```

FIGURE 4.9:

IMP OUTPUT, INTEGRATED CONTINUOUS EMPIRICAL DISTRIBUTION, DATA OF FIGURE 4.8

either (a) into an IMP.DAT data file (see Figure 4.8) somewhat similar to that discussed in Section 4.9 or (b) entered directly via the keyboard. The first line of the data file contains descriptive information (up to 60 characters) used to identify the file. (This line is mandatory.) The pairs of (X,Y) values are placed in subsequent lines of the file with the X-value starting in column 1 (decimal point required) and the Y-value following but separated by a comma (decimal point required). The end-points of the distribution (i.e., those with values of Y equal to zero) must be included in this data set. Since 15 vertical lines were drawn on Figure 4.7, the data file contains 17 entries after the first line. Data entries from the keyboard follow the same rules.

The results of selecting IMP option 2 on the Main Menu (and using the data file shown in Figure 4.8) are shown in Figure 4.9. IMP integrates the discrete data entered, and then prints a histogram of the integrated distribution. Finally, it prepares the cumulative probability distribution required for the MOUSE ECTS function.

EXERCISES

Which of the following are valid probability function statements and calls?

1. ECTS(R1,17)
2. EBETA(200.,100.,400.)
3. ECTS(D1,3.)
4. TRA(5.,7.,10.,20.)
5. BIN(0.3,10.)
6. GAU(-15.,5.)
7. GAUL(-15.,5.)
8. AGAUL(22.,30.)
9. TRI(5.,10.,20.)
10. READ(D7,30)

Find the errors, if any, in the following Empirical Data line sets.

(a) The function is EDIS(D1,3)

11.

```
PRICE...   PRICE OF ITEM
10.,0.3
20.,0.7
30.,1.0
```

12.

```
          CONTAM        FEED CONTAMINANTS
          1.,0.0
          2.,0.5
          3.,1.0
```

(b) The function is ECTS(D1,3)

13.

```
          PRICE       PRICE OF ITEM
          10.,0.3
          20.,0.7
          30..1,0
```

14. Construct a truncated triangular distribution with modal value 20, and lower and upper limits 10 and 70, respectively. The truncation limits are 15 and 60.

15. Production on three shifts is normally distributed with mean/standard deviations of 100/10, 75/5, and 55/5, respectively. Construct a composite probability function for daily production.

16. Fit the following two data sets to a classical probability distribution.

Data Set 1 (continuous data)		Data Set 2 (discrete data)	
48.87	71.20	4	1
60.63	75.99	7	5
88.99	37.68	1	5
143.06	80.46	14	4
19.31	135.46	6	5
145.66	93.72	7	5
165.16	60.97	6	3
34.17	253.71	2	2
78.40	219.31	7	6
42.53	121.76	6	8
123.09	144.76	8	3
47.72	165.64	10	7
76.65	223.06	5	5
81.56	78.71	3	2
99.67	21.99	7	6
28.77	25.83	3	4
33.81	55.57	6	7
109.39	292.71	5	7
290.52	16.82	5	2
169.18	137.94	3	5
68.63	372.50	4	5
303.63	44.01	8	4
33.27	48.09	7	6
306.19	160.71	3	4
20.15	82.71	5	1

17. Using IMP, integrate the empirical continuous distribution shown in Figure 4.10 using (a) 13 points and (b) using 25 points. Does the 25-point integration provide a substantially better fit than the 13-point integration?

REFERENCES

Burr, I.W., "A Useful Approximation to the Normal Distribution Function, With Application to Simulation", Technometrics, Vol. 9. No. 4, p. 647-651 (1967).

Foutz, R.V., "A Test For Goodness-Of-Fit Based Upon An Empirical Probability Measure", The Annual of Statistics, Vol. 8, No. 5, p. 989-1001 (1980).

Marsaglia, G. and Bray, T.A., "A Convenient Method for Generating Normal Variates", SIAM Review, 6, p. 260-264 (1980).

Naylor, T. J., Balinfty, J.L., Burdick, D.S., and Chu, K., Computer Simulation Techniques, John Wiley & Sons, New York, p. 1-352 (1966).

Peizer, D.B., and Pratt, J.W., "A Normal Approximation for Binomial, F, Beta, and Other Common, Related Tail Probabilities, I", J. Amer. Statistical Association, Vol. 63, No. 324, p. 1416-1456 (1968).

Pritsker, A.B., and Pegden, C.D., Introduction to Simulation and SLAM, John Wiley & Sons, New York, p. 567 (1979).

Von Neumann, J., "Various Techniques Used in Connection with Random Digits", Monte Carlo Method, Nat. Bur. of Standards, Applied Math. Series 12, p. 36-38 (1951).

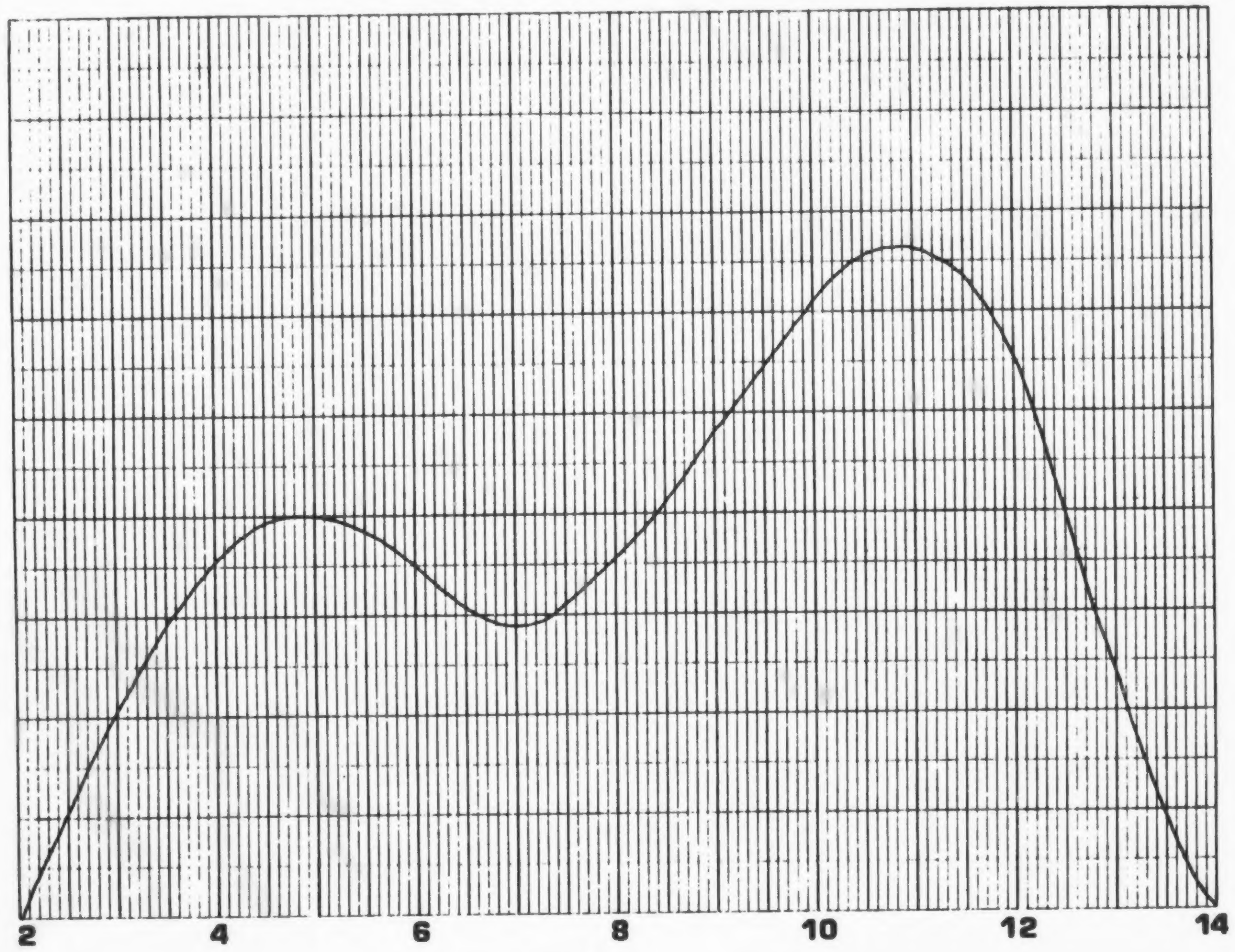


FIGURE 4.10:

EMPIRICAL CONTINUOUS PROBABILITY
DISTRIBUTION FOR EXERCISE NUMBER 17

CHAPTER 5

MOUSE LIBRARY INTEREST FUNCTIONS

5.1 Introduction

Economic analyses invariably involve interest formulas of various types, some of them unfortunately rather complex. For this reason, MOUSE provides eleven interest functions that may be called by you when desired. These functions cover almost all of the interest situations encountered in practice. Unlike stochastic input variables, the method of introduction of interest functions is direct, i.e., they are used directly in the risk equations. The general form of compound interest and gradient interest functions in MOUSE is:

function(interest rate,term,period)

Explanation of the period will be deferred until Section 5.4. For the time being, the period will be set to zero in the examples.

Decimal points are used in all interest functions. The functions are divided into three groups: (a) compound interest, (b) gradient series, and (c) continuous compounding. In the following formulas, the symbol P generally refers to a present worth, F refers to a future worth, and A refers to an annual payment or annuity. The basic payment or receipt involved in the formulas is the unit amount; for other quantities the function is merely multiplied by the actual amount involved.

5.2 Compound Interest

a. Single Payment Compound Amount Factor (SCA)

The formula for the compound amount, F_1, obtainable in n years at an interest rate of i from a single initial deposit of $1 is

$$F_1 = (1 + i)^n \tag{5.1}$$

This is called by the function, SCA(i,n,0.), where i represents the interest rate per interest term and n represents the number of interest terms. This convention is followed for all MOUSE interest functions. Example: If \$1 is invested at 6% compounded annually for 10 years, how much will be accumulated at the end of the term? Using SCA(0.06,10.0,0.), MOUSE will return the value of 1.79085.

b. Single Payment Present Worth Factor (SPW)

The formula for the present worth, P_1, of \$1 n years hence at an interest rate of i is

$$P_1 = 1/(1 + i)^n \qquad [5.2]$$

This is called by the function, SPW(i,n,0.). Example: What is the present worth of \$1, 5 years hence, if the interest rate is 10%? Using SPW(0.10,5.0,0.), MOUSE will return the value of 0.62092.

The two preceding formulas involved single payments. The next four involve uniform end-of-year payments.

c. Uniform Series Present Worth Factor (UPW)

The formula for the present worth, P_2, of \$1 deposited at the end of every year for n years at an interest rate i is

$$P_2 = [(1 + i)^n -1]/i(1 + i)^n \qquad [5.3]$$

This is called by the function UPW(i,n,0.). Example: What is the present worth of \$1 deposited at the end of each of 9 years at an interest rate of 20%? Using UPW(0.20,9.0,0.), MOUSE will return the value of 4.03097.

d. Uniform Series Present Worth Factor With Constant Percentage Change (UPWP)

The previous factor involved \$1 deposited at the end of every year. If each year after the first this amount is increased or decreased by 100j percent of the previous year's amount, the present worth, P_3, becomes

$$P_3 = [i/(i - j)][1 - (1 + j)^n(1 + i)^{-n}] \qquad [5.4]$$

This is called by the function UPWP(i,n,j,0.). Example: If \$1 is deposited at the end of the first year, and subsequent year-end deposits are increased 12% per year, what is the present worth at the end of 3 years at an interest rate of 8%. Using

UPWP(0.08,3.0,0.12,0.), MOUSE will return the value of 2.88194. Compare this result to that obtained when the change is a yearly 12% decrease: UPWP returns a value of 2.29512.

e. Uniform Series Compound Amount Factor (UCA)

The formula for the compound amount, F2, obtainable in n years from a series of end-of-year payments of $1 at an interest rate of i is

$$F_2 = [(1 + i)^n - 1]/i \qquad [5.5]$$

This is called by the function UCA(i,n,0.). Example: How much will be accumulated in a fund, earning 6% interest, at the end of 10 years if $1 is deposited at the end of each year? Using UCA(0.06,10.0,0.), MOUSE will return the value of 13.18084.

f. Sinking Fund Factor (SINK)

The formula for the uniform annual series of deposits, A_1, to produce $1 at the end of n years at an interest rate of i is

$$A_1 = i/[(1 + i)^n - 1] \qquad [5.6]$$

This is called by the function, SINK(i,n,0.). Example: How much must be deposited each year for 7 years at 8% interest in order to accumulate $1? Using SINK(0.08,7.0,0.), MOUSE will return a value of 0.11207.

g. Capital Recovery Factor (CAPREC)

The formula for the deposit, A_2, necessary to draw out $1 at the end of each year for n years, leaving nothing in the fund at the end, is

$$A_2 = [i(1 + i)^n]/[(1 + i)^n - 1] \qquad [5.7]$$

The Capital Recovery Factor is also the uniform end-of-year payment necessary to repay a debt of $1 in n years at an interest rate i. The function is called by CAPREC(i,n,0.). Example: What equal year-end withdrawals can be made each year for 10 years, leaving nothing in the fund after the tenth withdrawal, at an interest rate of 6% and an initial deposit of $1? Using CAPREC(0.06,10.0,0.), MOUSE will return the value of 0.13587.

5.3 Gradient Series

Sometimes receipts or expenses increase or decrease each year by varying amounts. If the change is the same each year, it is known as a uniform arithmetic gradient. Calculations involving gradients are most easily handled by first converting them to equivalent uniform series.

a. Gradient Series Equivalent Uniform Annual Amount Factor (GCA)

Using end-of-the-year payments, we assume that the payment the second year is greater than the first year by \$1, the third is \$1 greater than the second year, etc. The formula for the equivalent uniform annual figure, A_3, for n years at an interest rate of i is

$$A_3 = 1/i - (n/i)[i/(1 + i)^n - 1] \qquad [5.8]$$

This is called by the function GCA(i,n,0.). Example: There is an increase in cost of an item of \$1 per year over a term of 6 years; the interest rate is 12%. What is the equivalent uniform annual cost? Using GCA(0.12,6.0,0.), MOUSE will return the value of 2.17207.

b. Gradient Series Present Worth Factor (GPW)

The formula for the present worth of a gradient series, P_4, is merely the previous formula, A_3, multiplied by the uniform series present worth factor, P_2, i.e.,

$$P_4 = A_3 P_2 \qquad [5.9]$$

This is called by the function GPW(i,n,0.). Example: What is the present worth of the gradient series in the last example? Using GPW(0.12,6.0,0.), MOUSE will return the value of 8.93027.

5.4 Nominal Interest Rates

Although the interest period used in illustrating the above nine compound interest formulas was the year, any interest period could have been used. However, when other than annual compounding is used, a real difference exists between nominal and true (sometimes called " effective") interest rates. The MOUSE functions so far described are assumed to be provided with the true or effective rates. For example, to compute the compound amount of \$1 (single payment) for 8 years at a true annual interest rate of 6% compounded semiannually, we note that the number of periods is 8(2) = 16 and the rate per interest period is 6/2 = 3%. Thus, using SCA(0.03,16.0,0.), MOUSE will return the value of 1.60471.

If, on the other hand, a nominal annual interest rate, r, of 6% was given we would first have to convert it to an effective rate per annum, i, by the following formula:

$$i = (1 + r/m)^m - 1 \qquad [5.10]$$

where m is the number of times per year the amount is compounded. Thus, for an r of 6% in the above example,

$$i = (1 + 0.06/2)^2 - 1 = 0.0609.$$

Using SCA(0.0609,8.0,0.), MOUSE will return the value of 1.61596.

However, to spare you from having to make this sort of calculation, a provision has been made to use the compound interest and gradient series functions directly even when nominal annual interest rates are specified. To do so, the nominal interest rate is used and the "period" argument is set equal to the number of yearly compounding periods. For example, for the above problem in which the compounding is semiannual for 8 years at a nominal rate of 6%, the interest function would appear in the risk equations as SCA(0.06,8.0,2.0). The value of SCA would be returned as 1.61596. Note that for true interest rates, the value of the period is always zero.

Thus, MOUSE interest functions can be used readily for any compounding period, regardless of whether true or nominal interest rates are specified.

5.5 Continuous Compounding

All of the interest functions described so far have assumed that cash flow occurring throughout the year was concentrated at the end of the interest periods. An alternative convention is to assume that cash flow occurs uniformly throughout the year. To do so, however, it is necessary to use continuous compounding interest formulas. The general form of a continuous compounded interest function in MOUSE is:

function([-]interest rate,term)

The minus sign before the interest rate is optional and will be explained later.

a. Single Payment Present Worth Factor (CSPW)

The formula for the present worth, P_5, of \$1 for the period of time n-1 to n at an interest rate of i, is

$$P_5 = [(e^i - 1)/e^{i(n-1)}]ie^i \qquad [5.11]$$

This is called by the function CSPW(i,n). Example: What is the present worth of \$1 compounded continuously from the period year 3 to year 4 at an interest rate of 25%? Using CSPW(0.25,4.0), MOUSE will return the value of 0.45890.

b. Uniform Series Present Worth Factor (CUPW)

The formula for the present worth, P_6, of \$1 deposited continuously throughout the year for n years at an interest rate of i is merely the sum of the P_5 factors from year 1 to year n, i.e.,

$$P_6 = \sum_{j=1}^{n} P_{5j} \qquad [5.12]$$

This is called by the function CUPW(i,n). Example: What is the present worth of \$1 per year flowing continuously for 4 years at an interest rate of 25%? Using CUPW(0.25,4.0), MOUSE will return the value of 2.64583.

5.6 Nominal Interest Rates, Continuous Compounding

In the above two continuous interest functions, the interest rate is assumed to be the true or effective rate. As with the compound interest and gradient series functions, however, it may happen that the nominal interest rate, r, is specified instead of i. (In continuous compounding, the nominal annual rate is also known as the force of interest.) To convert a nominal rate to an effective rate, the following formula can be used:

$$i = e^r - 1. \qquad [5.13]$$

For example, a nominal rate of 25% is converted to an effective or true rate as follows:

$$i = e^{0.25} - 1 = 0.28403.$$

Using CSPW(0.28403,4.0), MOUSE will return the value of 0.41795. Again, however, in order to spare you from having to make this sort of calculation, a minus sign is used before the interest rate and the nominal interest rate is used. As an example, for the above problem in which the compounding is continuous with a nominal interest rate of 25% from the year 3 to year 4, the value of CSPW(-0.25,4.0) will be returned as 0.41795, the same value as shown with the additional hand calculation. Thus, the MOUSE continuous interest functions can also be used readily regardless of whether true or nominal interest rates have been specified.

For additional detail on the genesis and use of most of the interest formulas described here (the Uniform Series Present Worth Factor With Constant Percentage Change was developed by the author), it is suggested that the reader consult Grant and Ireson (1970).

5.7 Example of Probability Function Calls and MOUSE Interest Call Syntax

The following line was taken from the Waste Oil Reclamation Problem shown in Figure 3.2:

```
PVALUE=ANETB*UPW(0.08,XLIFE,0.0)-XININV
```

The function, UPW(0.08,XLIFE,0.0), is the Uniform Series Present Worth Factor with a <u>nominal</u> interest rate of 8%. The period varies from iteration to iteration with the variable XLIFE, i.e., project life is a stochastic variable. Note that even the simplest of problems involving uncertain project periods are almost impossible to calculate by hand, MOUSE removes this difficulty entirely.

A utility program, CHECKER (see Chapter 11, Section 11.2), can be used to examine a MOUSE source program to insure that: (1) the proper number of arguments are made to MOUSE interest calls, and (2) the arguments are reasonable (e.g., no negative interest rates, initial investments, or interest periods), the proper form of argument is used, i.e., decimal versus integer number or variable).

EXERCISES

Which of the following are valid interest function statements and calls?

1. SPW(.09,19,0.)
2. CAPREC(.12,3.,0.)
3. GCA(.20,10.0,0.)
4. SINK(10.,.10,0.)
5. UPWP(.09,.10,20.,0.)
6. CUPW(0.08,-15.,0.)
7. CSPW(0.08,15.,1.)
8. GPW(-.08,15.,0.)
9. SCA(.18,3.5,0.)

REFERENCE

Grant, E.L., and Ireson, W.G., Principles of Engineering Economy, Fifth Edition, The Ronald Press Co., New York, p. 1-640 (1970).

CHAPTER 6

THE MOUSE TABLE

6.1 Introduction

The MOUSE Table is the most important utility for any MOUSE program since it provides all of the vital statistics needed for the final output stochastic variables and for intermediate output stochastic variables as well. The MOUSE Table not only includes traditional statistics such as mean, standard deviation, minimum, maximum, and coefficient of variation (the coefficient of variation is equal to the standard deviation divided by the mean, and is a measure of relative variability of the variable), but it also calculates the average of values less than zero. If the random variable tabulated is in dollars, the average of values less than zero is the familiar "average opportunity loss" of economists or "expected value of perfect information" of decision scientists (see Winkler, 1972).

The MOUSE Table produces much more than just these statistics, however. It also produces the complete probability distribution and cumulative probability distribution of the variable. Thus you can find the probability of the random variable equal to or falling below a stated value, or the probability of the random variable equal to or falling above a stated value. The distributions are given in both histogram and graphical form, providing the advantages of both numeric and graphical presentation. Furthermore, it is not necessary to interpolate within MOUSE tables for the value of the variable at rounded fractiles of cumulative probability. A short table is printed out after the histogram giving the variable values at the following cumulative probabilities (in percentages):

0.1,1,5,10,25,50,75,90,95,99,99.9

Although the request for table output is made in the Program Block of a MOUSE program, every table (except for Dummy Tables - see Section 6.5) must be dimensioned; accordingly, a table involves two of the four basic MOUSE blocks: the Dimension Block and the Program Block.

Requests for MOUSE tables are of two basic forms: MOUSE-dimensioned and User-dimensioned. To structure a histogram properly, you must have a good estimate of the minimum and maximum values of the distribution. In complex problems, however, this information is usually not available unless a MOUSE run has already been made. Thus, if you dimension the histogram, two MOUSE runs must be made to obtain a usable histogram. An alternative is to take advantage of this option whereby MOUSE does the dimensioning automatically. Since it is simpler, the MOUSE-Dimensioned Table will be discussed first.

6.2 The MOUSE-Dimensioned Table

A MOUSE-dimensioned Table Output line has the following form:

```
CALL TABLE(variable,'(lname)',d1,d2,ni,ti)
```

where "variable" is the name of the variable to be tabulated, "lname" is an alphanumeric name (up to eight characters) for the table (note that this is a literal name and the single quotes and the parentheses must accompany the name exactly in the order shown) and can contain up to but not more than eight characters), d1 and d2 are dummy floating point variable names chosen arbitrarily (just remember that d1 and d2 cannot have been used elsewhere in the program), "ni" (an integer) is the number of histogram intervals desired for the table output (including a possible underflow and/or overflow interval (in grouping data into frequency distributions, a rough working rule used by statisticians is to aim to have about 20 intervals), and "ti" is the name of an array used for storing the histogram table between iterations of the program. Also note that the variable name must be a decimal name. (A decimal name is one that starts with any letter except I, J, K, L, M, or N. See Chapter 3, Section 3.2).

To use this option, a minimum of 100 iterations (ITER=100) must be used. As an example, the following is the Table Output line of Figure 3.2:

```
CALL TABLE(PVALUE,'(PRES VAL)',DUM0,DUM1,40,TAB)
```

With one exception (the Program Listing line; see Chapter 7, Section 7.10 the last executable statement in any MOUSE program (i.e., the last user-written line before MOUSE Control line Number 5) must be a Table Output line, although additional Table Output lines many also be used at any appropriate place in the Program Block. Each program must also contain one or more Table Dimension lines that dimension the arrays used in the Table Output lines (an exception is when a Dummy Table is used; see Section 6.5). These have the following form:

```
DIMENSION t1(n1),t2(n2),etc.
```

where the "ti" arguments are identical to those specified in their respective Table Output lines and the "ni" arguments are equal to the respective ni plus six. These additional six elements of the dimension number are used to accumulate the mean, standard deviation, minimum, maximum, and percentage of entries less than zero. For this reason, the minimum dimension of any table is six. In the Waste Oil Reclamation Problem shown in Figure 3.2, the Table Dimension line was:

```
DIMENSION TAB(46)
```

The Table Output line is placed after (usually immediately after) the uncertainty equation that calculates the quantity to be tabulated; all Table Dimension lines are placed in the Dimension Block of the MOUSE program. (See Chapter 11, Section 11.6, for a discussion of TRAP, a MOUSE utility that greatly facilitates the preparation of the Dimension Block and even of the Table Output lines themselves. Also, CHECKER, Chapter 11, Section 11.2, will examine a MOUSE source file to insure that arguments of standard and dummy tables are correct, and that the dimension of the array used for storing the standard histogram table agrees with those specified in the Dimension Block.)

The algorithm used in MOUSE for the MOUSE-dimensioned histogram guarantees (see Grant and Leavenworth, 1972) that:

a. If the variable tabulated is distributed normally, approximately 0.05% of the values will fall into the underflow/overflow portions of the histogram;
b. If the variable is distributed unimodally and symmetrically about the mean, less than 3.6% of the values will fall into the underflow/overflow portions (Camp-Meidell inequality); and
c. Regardless of the distribution of the variable, less than 8.2% of the values will fall into the underflow/overflow portions (Tchebycheff Inequality).

6.3 The User-Dimensioned Table

The user-dimensioned Table Output line has the following form:

```
CALL TABLE(variable,'(lname)',lower,interval,-ni,ti)
```

where "variable", "lname", "ni", and "Ti" are defined as for MOUSE-dimensioned tables. Note, however, that ni is preceded by a minus sign. The dummy floating point variable names are replaced in the User-dimensioned table by "lower", which is the lower bound of the histogram to be produced by MOUSE, and "interval" (a decimal number greater than zero), which is the width of the intervals desired. We could have, for example, replaced the Table Output line of Figure 3.2 with:

```
CALL TABLE(PVALUE,'(PRES VAL)',-3500.,250.,-40,TAB)
```

This would start the lower interval at PVALUE equal to -3500, and make the width of each histogram interval equal to 250. Like MOUSE-dimensioned tables, the appropriate Table Dimension lines must be used with User-dimensioned lines also.

6.4 The Short Table

A "short" version of the Table Output line can be used that produces only the six basic MOUSE table statistics, i.e., mean, standard deviation, minimum, maximum, coefficient of variation, and average of values less than zero, skipping the histogram and graphs. The short version is of the form,

```
CALL TABLE(variable,'(lname)',0.,0.,0,Ti)
```

In this Output line, the number of histogram intervals is given as zero, and "lower" and "interval" are given as zeros with decimal points. A Table Output line with dimension equal to 6 will, of course, still have to be supplied.

6.5 The Dummy Table

It has previously been mentioned that, with the exception of the Program Listing line (see Chapter 7, Section 7.10), the last executable statement in any MOUSE program must be a Table Output line. However, when you do not want any table output (e.g., when generating random data for testing purposes - see Appendix E - or when you just desire Plot output - see Chapter 8), a Dummy Table can be substituted. The Dummy Table Output Line is of the form:

```
CALL DTABLE(switch)
```

where "switch" is either "1" or "2". The usual default for the "switch" is "1" in which case the standard MOUSE output file, MOUSE.OUT, will be constructed. With "switch" set to "2", the MOUSE.OUT file will be deleted at the end of the run. Note that Table Dimension lines are not used with Dummy Tables.

6.6 User Table Scaling

The table subroutine in MOUSE involves the calculation of the standard deviation of the variable involved, a matter of squaring and summing the values of the variable over the number of iterations used. If these values are large, the combination of squaring and adding over many iterations could easily exceed the capacity of the computer. When this happens, most computers will print a string of asterisks in lieu of the result or else print some form of error message such as "FLOATING POINT ARITHMETIC OVERFLOW". Although some automatic scaling is built into MOUSE, it may occur that the computer capacity is exceeded. In such a case, you may try another run, scaling the variable right in the Table Output line. For example, to scale down the value of PVALUE by 1000 in Figure 3.1, the Table Output line would be written as:

```
CALL TABLE(PVALUE/1000.,'(PRES VAL)',DUM0,DUM1,40,T1)
```

You must remember, however, that all of the numbers printed in the table so produced will also be scaled by this factor of 1000 (except the Coefficient of Variation which has no absolute scale). User-scaling may be applied with any kind of MOUSE table (user- or MOUSE-dimensioned and short forms). Scaling in a table does <u>not</u> affect the value of the variable for use in subsequent calculations in a model. Note that the scaling factor must have a decimal point.

6.7 Literal Names in MOUSE

Literal names are names that ultimately are reported in MOUSE output reports. Such names are <u>always</u> first enclosed in parentheses, then enclosed in single quotes, e.g.

```
'(COST)'
```

This may appear to be somewhat awkward, but it necessary in order to accommodate different FORTRAN compilers that may be used to implement MOUSE. Literal names are found in the following MOUSE output lines:

Name line
Plot Output Line
Program Listing line
Table Output line
Trace line
Vector Table Output line

6.8 Mixing Table Types

Tables of different types (i.e., user-dimensioned, MOUSE-dimensioned, and short tables) can appear in a single program.

EXERCISES

Which of the following table statements are valid?

```
1.    CALL TABLE(COST,’(COST)’,1000.,20.,-40,T1)
2.    CALL TABLE(REVENUE,’(REVENUE)’,A1,A2,40.,T1)
3.    CALL TABLE(WAGECOST,'WAGECOST',5.5,0.6,-40,T1)
4.    CALL TABLE(X5,’(COST FAM)’,-450.,20.,-60,T1)
5.    DIMENSION T1(40)
6.    CALL TABLE(CUSTOMER,’(CUSTOMER)’,AT1,BT1,-40,T1)
7.    CALL TABLE(SPEED/1000,’(SPEED)’,Z1,Z2,100,T1)
8.    CALL TABLE(X5,’(X5)’,100.,0.,-20,T1)
9.    DIMENSION (T1,40)
10.   CALL TABLE(WAGE,’(WAGE)’,0.,0.,20,T1)
11.   DIMENSION R2(83)
```

REFERENCES

Grant, E.L., and Leavenworth, R.S., *Statistical Quality Control*, McGraw-Hill Book Co., New York, p. 74-75 (1972).

Winkler, R.L., *Introduction to Bayesian Inference and Decision*, Holt, Rinehart and Winston, New York, p. 1-563 (1972).

CHAPTER 7

USER CONTROL DECISIONS

7.1 Introduction

As was mentioned in Chapter 3, the Specification Block (i.e., those lines placed immediately after MOUSE Control Line 3) informs the computer of certain of your control decisions. Five line types to be discussed in this Chapter and one in the next Chapter will comprise the types to be found in the Specification Block. This Chapter will also explain the use of the Name Line and how multiple runs (i.e., runs in which parameters are altered from run to run) are made in MOUSE.

7.2 The Iteration Number Line

The number of iterations within a MOUSE program is determined by the Iteration Number Line. This line is of the following form:

```
ITER=n
```

where n is the number of iterations desired (an integer). Usually this is a fairly substantial number (1000 or greater), and normally never lower than 100 (it can not be less than 2). If you do not supply an Iteration Number line, the default number of iterations is 1000.

7.3 The Snap Interval Line

There are two types of Monte Carlo simulations with respect to the number of iterations in a run (Law, 1980). A steady state simulation is one for which the output is conceived as a limit as the number of iterations becomes infinite. From a practical standpoint, the actual number of iterations in a steady state simulation is made large enough to get "good" estimates of the output quantities of interest. The other type is the terminating simulation where the outputs are defined relative to an interval of simulated time, $[0,T_e]$, where T_e is the instant in the simulation when a specified event, E, occurs. (T_e may be a stochastic variable.) The event E is specified before the simulation begins. Since the outputs for terminating simulations explicitly depend on the state of the simulated system at time 0, care must be taken in choosing the initial conditions. (An example of a terminating simulation is that of simulating a frame in bowling. The frame is over when either two balls are thrown or when all the pins go down, whichever comes first. The end of the frame is E, and the time at the end of the frame is T_e. The initial conditions, as far as score is concerned, are represented by the history of the prior frame. If, for example, a spare was achieved in the prior frame, then the number of pins downed on the first ball are also added to the score for the preceding frame.)

The Monte Carlo uncertainty analysis simulations dealt with in this book are of the steady state type; initial conditions, therefore, are of no concern. As it is sometimes difficult to decide how many iterations should be employed to achieve steady state conditions in a Monte Carlo simulation, MOUSE provides an option called a "snap" which is used in conjunction with MOUSE tables. The Snap Interval Line is of the form:

```
ISNAP=s
```

where s is some integer smaller than the n of the Iteration Line. In Figure 7.1, which is Modification #1 of the Waste Oil Reclamation Problem, the use of the Snap line is illustrated. The snap interval is set at 1000, and the iteration line is set at 5000. This will produce a total of four snap outputs, i.e., at number of iterations = 1000, 2000, 3000, and 4000. At 5000, the usual MOUSE output will be delivered. The snap output for the program of Figure 7.1 is shown in Figure 7.2. (The snap output follows the echo print of any input data you may have supplied; otherwise, the snap output is the first thing printed by MOUSE.) At each snap interval the mean and standard deviation is printed out for each table defined in the program. You can follow the snap output to see whether the output has reached steady state. At number of intervals = 4000, it does not appear from Figure 7.2 that steady state has been reached.

7.4 The Seed Reset Line

For those who like to rerun programs using different random number seeds in order to see what effect a different set of random numbers has on the output, a Seed Reset Line is available. This line is of the following form:

```
IRAN=seed
```

where "seed" is any odd integer not exceeding the integer capability of the computer being used.

```
C
C WASTE OIL RECLAMATION PROBLEM MODIFICATION #1
C
      DIMENSION D1(3,2),D2(4,2),D3(4,2)
C
C MATRIX DIMENSION LINE
C
      DIMENSION COEF(3,1)
C
      DIMENSION TAB(46)
      COMMON/Q1/C80/Q2/JUMP/Q3/LOOP/Q4/SENS/Q5/IDATA/Q6/IOUT/Q7/IPASS/Q8
     &/IRAN/Q9/IREAD/Q10/ISCR1/Q11/ISCR2/Q12/ISNAP/Q13/ITER/Q14/ITOG/Q15
     &/IWRIT/Q16/IXXX/Q17/OK1/Q18/OK2/Q19/OK3/Q20/OK4/Q21/OK5/Q22/OK6
C
C NAME LINE
C
      CALL NAME('(WASTE OIL RECLAMATION PROBLEM - MODIFICATION #1)')
C
      CALL READ(D1,3)
      CALL READ(D2,4)
      CALL READ(D3,4)
C
C MATRIX READ LINE
C
      CALL MREAD(COEF,3,1)
C
C SNAP INTERVAL LINE
C
      ISNAP=1000
C
C ITERATION NUMBER LINE
C
      ITER=5000
C
C PASS CONTROL LINE
C
      DO 15 IPASS=1,3
C
5     DO 10 IXXX=1,ITER
C
```

```
      CONTAM=DUNI(1.0,4.0)
      RAND1=CUNI(-10.0,10.0)
      PRICE1=EDIS(D1,3)
      SLOPE=ECTS(D2,4)
      PROD2=STEP(D3,4)
      RAND2=GAU(0.00,0.10)
      PRICE3=EBETA(6.5,8.0,9.5)
      PROD3=EXPON(18.0)
      XLIFE=DTRI(5.0,10.0,15.0)
      XININV=TRA(1600.0,2000.0,2200.0,2400.0)
C
      THRU1=10.0-CONTAM
      YIELD1=10.0-COEF(IPASS,1)*CONTAM
      PROD1=THRU1*YIELD1+RAND1
      GREV1=PRICE1*PROD1
      PRICE2=10.0-SLOPE*PROD2+RAND2
      GREV2=PRICE2*PROD2
      GREV3=PRICE3*PROD3
      ANETB=GREV1+GREV2+GREV3-70.0
      PVALUE=ANETB*UPW(0.08,XLIFE,0.0)-XININV
C
      CALL TABLE(PVALUE,'(PRES VAL)',DUM0,DUM1,40,TAB)
C
10    CONTINUE
      GOTO(5,15), JUMP
15    CONTINUE
      CALL LIST('(WO1.FOR)')
      END
```

FIGURE 7.1: WASTE OIL RECLAMATION PROBLEM, MODIFICATION #1

```
                    **********************************
                    *       SUMMARY OF SNAPS         *
                    *       PASS NUMBER    1         *
                    **********************************



SNAP AT NUMBER OF ITERATIONS =  1000     FOR QUANTITY PRES VAL
  MEAN =       2855.27000                   STANDARD DEVIATION =       1874.10900


SNAP AT NUMBER OF ITERATIONS =  2000     FOR QUANTITY PRES VAL
  MEAN =       2851.14600                   STANDARD DEVIATION =       1829.24000


SNAP AT NUMBER OF ITERATIONS =  3000     FOR QUANTITY PRES VAL
  MEAN =       2865.37400                   STANDARD DEVIATION =       1817.95600


SNAP AT NUMBER OF ITERATIONS =  4000     FOR QUANTITY PRES VAL
  MEAN =       2829.80000                   STANDARD DEVIATION =       1820.22100
```

FIGURE 7.2

SNAP OUTPUT FOR WASTE OIL RECLAMATION PROBLEM, MODIFICATION #1

7.5 The Output Destination and Column Output Lines

The destination of MOUSE output depends upon the logical unit or device number selected when modifying MOUSE to run on a specific computer system (the distribution version of MOUSE directs output to the file, MOUSE.OUT) For most purposes, it is probably best that output be directed to a file; such files are permanent (until you delete them) and can be read and printed at your leisure.

If it is desired that output for a particular run be directed to your terminal, however, this can be accomplished by use of an Output Destination Line (which is placed in the Specification Block) of the following form:

```
IOUT=IWRIT
```

Normally, all MOUSE output is dimensioned for 132 columns; if the device used cannot be set this wide, such as an 80-column video display terminal, then the output will "wrap around" and be difficult to read. You can, however, instruct MOUSE to redimension output to 80 columns by way of a <u>Column</u> <u>Output</u> <u>Line</u> of the following form:

```
C80=80
```

The 80-column output is not quite as elegant as the 132-column output, but the identical information is presented in either format. This line may be placed anywhere in the Specification Block.

7.6 Summary of the Specification Block

Except for the Sensitivity Curtailment line to be discussed in the next Chapter, all of the Specification Block lines have now been introduced and can be summarized as follows:

Line	Type
C80=80	Column Output
SENS=c	Sensitivity Curtailment (see Chapter 8)
IOUT=IWRIT	Output Destination
ISNAP=s	Snap Interval
ITER=n	Iteration Number
IRAN=seed	Seed Reset

All of these lines are optional (but note the default to 1000 if an ITER Line is not used). Note that TRAP (see Chapter 11, Section 11.6) can be used to prepare the Specification Block.

7.7 Multiple Runs: Matrices and Vectors

You may wish to vary one or more parameters of a model without having to make separate MOUSE runs. By "parameter" is meant a temporary constant as, for example, the coefficient of 2 in the YIELD1 equation shown in Figure 7.1, i.e.,

```
YIELD1=10.-2.0*CONTAM
```

The investigator might wish, say, to study the effects of changing this parameter from 2 to 1 and from 2 to 3. This is accomplished by placing the values 1, 2, and 3 into a matrix. A matrix is identified by the usual rules for naming variables in MOUSE, followed by a parenthetical expression containing two indices in integer form. The first index is a special one called "IPASS", e.g., COEF(IPASS,n), and the reason for this special index will be explained in the next Section. The second index is the number of parameters to be changed. In this example we are changing only one parameter, i.e., the coefficient in the YIELD1 equation noted above; hence, the second index is set equal to 1, i.e., COEF(IPASS,1).

Matrices must be dimensioned, a procedure similar to but slightly different from that used for the dimensioning of empirical variables where the second index is always 2. In matrices, the second index can be any integer. The <u>Matrix Dimension Line</u> has the form:

```
DIMENSION name(n,m)
```

where "name" is the name of the matrix, n is the number of rows, and m is the number of columns in the matrix. (See Chapter 11, Section 11.6, for a discussion of TRAP, a MOUSE utility that greatly facilitates the preparation of the Dimension Block.) Also note that the name must be a decimal name. (A decimal name is one that starts with any letter <u>except</u> I, J, K, L, M, or N. See Chapter 3, Section 3.2).

The Waste Oil Reclamation Problem shown in Figure 7.1 is, as has been mentioned, a modification of the original problem shown in Figure 2.1. In Figure 7.1 the slope coefficient of the YIELD1 equation has been modified as discussed above. Accordingly, a Matrix Dimension line was added to the program as follows:

```
DIMENSION COEF(3,1)
```

the 3 signifying that you are going to make three runs or "passes", each pass with a different value for the coefficient parameter, COEF. The YIELD1 equation was also modified in Figure 3.1 to:

```
YIELD1=10.-COEF(IPASS,1)*CONTAM
```

Note that, when used in an uncertainty equation, a matrix <u>must</u> use IPASS as its first index.

To read in the value of any matrix a <u>Matrix Read Line</u> of the following form is used:

```
CALL MREAD(name,n,m)
```

where the arguments are the same as discussed for the Matrix Dimension line. Decimal points are not used in Matrix Read lines.

The matrix data is treated much like empirical variable data in that it is entered free-form (in a MOUSE.DAT Data File), i.e., each row of the matrix is entered, starting at column 1, with the numbers separated by commas. Decimals are required in the numbers. If large matrices are involved, do not split a number at the end of a line but merely go to the next line and continue. Start new rows of the matrix on new data lines. One should not combine data from two separate rows in the matrix on one line.

The Matrix Data in the MOUSE.DAT file for this problem would appear as follows:

```
COEF...   COEFFICIENT IN YIELD1 EQUATION
1.
2.
3.
```

The first line is a name line (and is required) and may be punched in any column. The numbered order of the MOUSE passes corresponds to the rows of the matrix. Pass 3, for example, sets the value of COEF(IPASS,1) equal to 3.0. As in all data lines, decimal points are used.

Alternatively, we can bypass the Matrix Read Line by defining the matrix data directly in the Specification Block, e.g.,

```
COEF(1,1)=1.0
COEF(2,1)=2.0
COEF(3,1)=3.0
```

However, a 3 x 1 matrix (as is COEF) is identical to a 3-dimensional vector, which is a bit simpler. Vectors can be dimensioned using a <u>Vector</u> <u>Dimension</u> <u>Line</u> which has the following general form:

```
DIMENSION name1(n1)
```

where "namei" is the name of the ith vector and "ni" is its dimension. Also note that the name must be a decimal name. (A decimal name is one that starts with any letter <u>except</u> I, J, K, L, M, or N. See Chapter 3, Section 3.2). In this case, our information would be read in the Specification Block as follows:

```
COEF(1)=1.0
COEF(2)=2.0
COEF(3)=3.0
```

and the YIELD1 equation would be changed to:

```
YIELD1=10.-COEF(IPASS)*CONTAM
```

Note that again, the index used is "IPASS".

7.8 Multiple Runs: The MOUSE Pass

A <u>Pass</u> <u>Control</u> <u>Line</u> is required when implementing multiple runs. The line is of the form:

```
DO 15 IPASS=1,n
```

where "n" is defined as for the Matrix or Vector Dimension line. This line must be placed immediately before MOUSE Control Line 4.

If a matrix is used, by adding additional columns to the matrix you can alter more than one parameter during a pass. In the THRU1 and YIELD1 equations, for example, the parameter "10" is used. We could alter this parameter on different MOUSE passes by redefining it as COEF(IPASS,2), i.e.,

```
THRU1=COEF(IPASS,2)-CONTAM
YIELD1=COEF(IPASS,2)-COEF(IPASS,1)*CONTAM
```

Suppose we place the following Matrix Data in the MOUSE.DAT file:

```
COEF...   COEFFICIENTS IN THRU1 AND  YIELD1 EQUATIONS
1.,10.
1.,20.
2.,10.
2.,20.
```

This provides four MOUSE passes. Pass 3, for example, would set the intercepts in both equations equal to 10 and the slope in the second equation equal to 2. The Matrix Dimension line in this case would appear as:

```
DIMENSION COEF(4,2)
```

and the Matrix Read line would appear as:

```
CALL MREAD(COEF,4,2)
```

As can be seen, the use of the MOUSE pass permits you to make multiple runs, altering any combination of parameters of the model desired. A schematic outline of the lines involved in a typical MOUSE pass is shown in Figure 7.3, and a small portion of the table output from the program of Figure 7.1 is shown in Figure 7.4 where the effect of the COEF variable is very clear.

In addition to altering parameters, the MOUSE pass can also be used with IF statements to completely change the probability distributions of the model in multiple runs (see Chapter 14, Section 14.4). This adds very powerful sensitivity analysis capabilities to MOUSE.

```
[Dimension Block]                DIMENSION PAR(n,m)
                                   .
                                   .
                                   .
[Read Block]                     CALL MREAD(PAR,n,m)
                                   .
                                   .
                                   .
[PCC]                            DO 15 IPASS=1,n
[MCC 4]                 5        DO 10 IXXX=1,ITER
                                   .
                                   .
                                   .
[Program Block]                  ......=...PAR(IPASS,i)...
                                   .
                                   .
                                   .

[MCC 5]                 10       CALL END
[MCC 6]                          GOTO(5,15), JUMP
[MCC 7]                 15       CONTINUE
[MCC 8]                          END
```

Key: PCC = Pass Control Line
MCC = MOUSE Control Line
i = ith Parameter (i is a value between 1 and m)

FIGURE 7.3:

AN OUTLINE OF THE MOUSE PASS

7.9 Program Output Titles: The Name Line

Program titles, up to 60 characters in length, are placed in MOUSE output by use of a <u>Name Line</u> of the following form:

```
CALL NAME('(lname)')
```

where "lname" is a literal name descriptor of no more than 60 characters in length. Note that the descriptor must be enclosed in the single quotes and parentheses exactly as shown. The <u>Name Line</u> is placed <u>immediately</u> after Mouse Control Line # 3 and it is <u>mandatory</u>. The TRAP utility (see Chapter 11, Section 11.6) facilitates the preparation of this line.

```
                  ******************************************
                  *    DISTRIBUTION FOR QUANTITY PRES VAL  *
                  *           PASS NUMBER    1             *
                  ******************************************


NUMBER OF ITERATIONS =  5000         MEAN =     2815.72500             STANDARD DEVIATION =      1818.11100
                                  MINIMUM =    -1834.04600      COEFFICIENT OF VARIATION, % =      64.56989
                                  MAXIMUM =    11838.67000

AVERAGE OF VALUES LESS THAN ZERO =      -493.14260




                  ******************************************
                  *    DISTRIBUTION FOR QUANTITY PRES VAL  *
                  *           PASS NUMBER    2             *
                  ******************************************


NUMBER OF ITERATIONS =  5000         MEAN =     1876.57200             STANDARD DEVIATION =      1922.17400
                                  MINIMUM =    -3250.27700      COEFFICIENT OF VARIATION, % =     102.43010
                                  MAXIMUM =    10609.75000

AVERAGE OF VALUES LESS THAN ZERO =      -766.87190




                  ******************************************
                  *    DISTRIBUTION FOR QUANTITY PRES VAL  *
                  *           PASS NUMBER    3             *
                  ******************************************


NUMBER OF ITERATIONS =  5000         MEAN =      860.50080             STANDARD DEVIATION =      2151.11500
                                  MINIMUM =    -4596.54200      COEFFICIENT OF VARIATION, % =     249.98410
                                  MAXIMUM =     9859.02800

AVERAGE OF VALUES LESS THAN ZERO =     -1212.52300
```

FIGURE 7.4: Partial PASS Output For Waste Oil Reclamation Problem, Modification #2

7.10 The Program Listing Line

It is sometimes desirable to append a listing of your source program file (see Chapter 7, Section 7.10) to the MOUSE output file, MOUSE.OUT. This is accomplished by use of a <u>Program Listing Line</u> of the following form:

```
CALL LIST('(lname)')
```

where "lname" (a literal name enclosed in single quotes and parentheses as shown) is the name of the source file to be listed. An example is:

```
CALL LIST('(PROGRAM.FOR)')
```

The Program Listing line <u>must</u> be placed between MOUSE Control Lines numbers 7 and 8, e.g.,

```
15    CONTINUE
      CALL LIST('(PROGRAM.FOR)')
      END
```

If a non-existent file is referenced, MOUSE will report the fact and, of course, no listing will be produced.

7.11 Preparing and Checking Specification Block Lines

See Chapter 11, Section 11.6, for a discussion of TRAP, a MOUSE utility that greatly facilitates the preparation of the Specification Block lines discussed in this chapter. Also, CHECKER (Chapter 11, Section 11.2) will examine a MOUSE source file to insure that these lines are correct both in form and argument.

EXERCISES

Which of the following control lines are valid ?

1. ITER=2756
2. IRAN=37654
3. DIMENSION COEF(3,15)
4. YIELD=TAXES+SLOPE(2,1)-500.
5. CALL MREAD(REV,4)

What, if anything, is wrong with the following Pass Control line?

6. `DO 15 I=1,LIFE`

Find the errors, if any, in the following Matrix Data lines:

(a) The dimension statement is DIMENSION VELOC(2,3)

```
7.

          VELOC        COEFFICIENTS IN SPEED EQUATIONS
          10.,20.,30.
          5.,4.,1.
```

(b) The dimension statement is DIMENSION VELOC(4,2)

```
8.

          VELOC        COEFFICIENTS IN SPEED EQUATIONS
          10.,20.,5.,4.
          5.,4.,40.,100.
```

REFERENCE

Law, A.M., "Statistical Analysis of Simulation Output Data", *Proceedings of 1980 Winter Simulation Conference*, December 3-5, Orlando, Florida, p. 361-370 (1980).

CHAPTER 8

ADDITIONAL MOUSE FUNCTIONS

8.1 Introduction

This chapter discusses two MOUSE functions that expand the utility of the system. The first affords an easy way to obtain certain graphical plots of pairs of variables, and the second concerns sensitivity analysis.

8.2 The Plot Output Line

You might wish to know something about the functional relationship between two stochastic variables in a MOUSE program. If the two variables are output variables, there is no way for you to glance at the risk equations and easily deduce the nature of this relationship. Even if one of the variables was an input variable, there might be a considerable number of equations between the introduction of the input variable and the output variable of interest. Once again, it would be difficult to deduce the nature of the relationship simply by looking at the intervening equations.

MOUSE has an option whereby any variable can be plotted against another. In addition, a total of up to eight regression equations are automatically fitted to the plotted data. The Plot Output Line has the following form:

```
CALL PLOT(xvar,'(lxname)',yvar,'(lyname)',ti)
```

where the arguments are analogous to those defined for Table Output Lines. Note that "xvar" is the "independent" or exogenous variable and that "yvar" is the dependent or endogenous variable. Also note that both the xvar and yvar variable names must be decimal names. (A decimal name is one that starts with any letter except I, J, K, L, M, or N. See Chapter 3, Section 3.2). In Figure 8.1 (Modification #1 of the Regulatory Problem), a Plot Output Line has been used to plot FACTOR against DEPTHL, the latter being considered as the independent variable (which it certainly is in the Regulatory problem). This line appears in Figure 8.1 as follows:

```
CALL PLOT(DEPTHL,'(DEPTHL)',FACTOR,'(FACTOR)',P1)
```

```
C
C REGULATORY PROBLEM MODIFICATION #1
C
      DIMENSION TAB(46)
C
C PLOT DIMENSION LINE
C
      DIMENSION P1(100,3)
C
      COMMON/Q1/C80/Q2/JUMP/Q3/LOOP/Q4/SENS/Q5/IDATA/Q6/IOUT/Q7/IPASS/
     &/IRAN/Q9/IREAD/Q10/ISCR1/Q11/ISCR2/Q12/ISNAP/Q13/ITER/Q14/ITOG/Q
     &/IWRIT/Q16/IXXX/Q17/OK1/Q18/OK2/Q19/OK3/Q20/OK4/Q21/OK5/Q22/OK6
      CALL NAME('(REGULATORY PROBLEM  - MODIFICATION #1)')
      ITER=5000
5     DO 10 IXXX=1,ITER
      PRECIP=TRA(762.0,1016.0,1270.0,1524.0)
      RLEACH=CUNI(.5,1.0)
C
C SENSITIVITY CONTROL LINES
C
      CALL SEN(RLEACH,'(RLEACH)',50.,0.)
      CALL SEN(RLEACH,'(RLEACH)',0.,-50.)
C
      DEPTHL=TRA(3.0,6.1,9.1,15.2)
      WDENS=TRI(297.0,415.0,534.0)
      FACTOR=70000*(PRECIP*RLEACH)/(DEPTHL*WDENS)
C
C PLOT OUTPUT LINE
C
      CALL PLOT(DEPTHL,'(DEPTHL)',FACTOR,'(FACTOR)',P1)
      CALL TABLE(FACTOR,'(FACTOR)',DUM0,DUM1,40,TAB)
10    CONTINUE
      GOTO(5,15), JUMP
15    CONTINUE
      CALL LIST('(REG1.FOR)')
      END
```

FIGURE 8.1:

REGULATORY PROBLEM MODIFICATION #1

Along with the Plot Output line the program must contain a <u>Plot</u> <u>Dimension</u> <u>Line</u> of the following form:

DIMENSION T1(100,3),T2(100,3),T3(100,3), etc.

(See Chapter 11, Section 11.6, for a discussion of TRAP, a MOUSE utility that greatly facilitates the preparation of the Dimension Block. Also, CHECKER, Chapter 11, Section 11.2, will examine a

MOUSE source file to insure that arguments of Plot Output lines are correct, and that the dimension of the array used for storing the plot table agrees with that specified in the Dimension Block.) Also note that the variable names T1, T2, etc., must be decimal names. (A decimal name is one that starts with any letter except I, J, K, L, M, or N. See Chapter 3, Section 3.2).The Plot Dimension line in Figure 8.1 is, for example,

```
DIMENSION P1(100,3)
```

The plot output for the program of Figure 8.1 is shown in Figure 8.2. The points represent the average value of FACTOR for the intervals shown on the DEPTHL axis. (They are plotted at the midpoints of the intervals of the DEPTHL axis, however, rather than the lower bound of the intervals as is done in MOUSE tables.) Accordingly, the points represent different numbers of observations. In order not to mislead the eye, points that represent less than 20% of the maximum number of observations represented by any one point are plotted as a "+" character; all other are plotted as an "*". The dotted line represents the best fit of eight models: linear, quadratic, cubic, reciprocal, square root, logarithmic, power, and exponential. Underneath the graph in Figure 8.2 is a summary of these fitted equations (fitted via a weighted regression technique). The values of these parameters are given, as well as the goodness-of-fit as measured by R-square (the coefficient of determination). Note, however, that some equations cannot be fitted to data sets having zero or negative numbers. One cannot, for example, take the square root of a negative number. In such cases. the inappropriate equations will automatically be deleted from the regression summary.

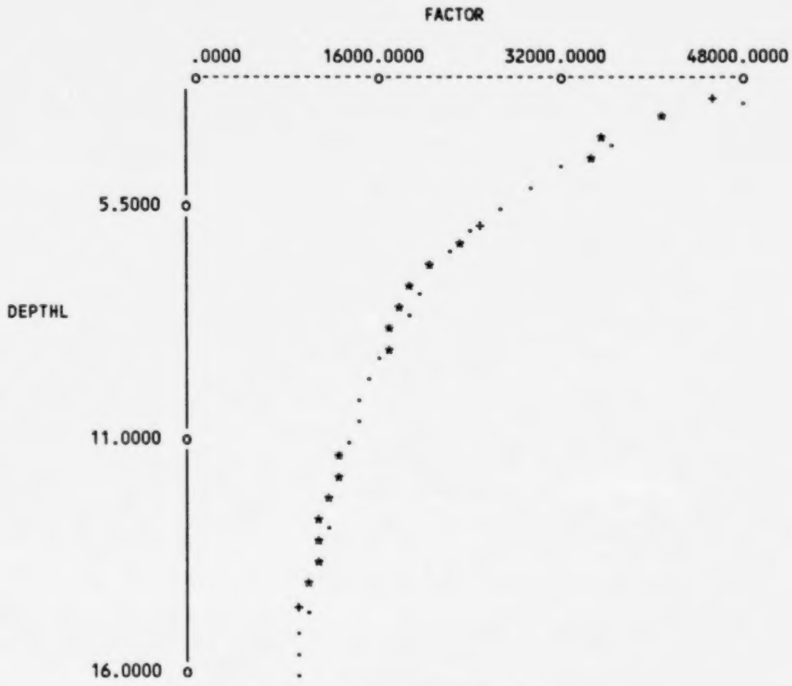
In Figure 8.2, we find that the best-fitting equation is the cubic (R-square = 0.56), the equation being

$$Y = 3743 - 0.0042X^3$$

where Y is FACTOR and X is DEPTHL (the E+04 and E-02 in the line for the cubic equation in Figure 8.2 stand for multiplication by 10^4 and 10^{-2}, respectively). There were no zero values for X nor were there negative values for X or for Y; accordingly, all eight equations were fitted.

As with Table Output Lines, you can scale the variables in a Plot Output Line (see Chapter 8, Section 8.2). The scales of the plot axes and the regression parameters, of course, will reflect this scaling. Also, Plots require a minimum of 100 iterations. Fewer than this will result in termination of the run. As has been mentioned before (Chapter 6, Section 6.2), other than a Program Listing Line the last executable statement in a MOUSE program must be a Table Output Line; it cannot be a Plot Output Line.

```
***************************************************
*    PLOT OF QUANTITY FACTOR   VERSUS DEPTHL      *
***************************************************
```



```
Point code: * = within 20% of maximum weight
            + = less than 20% of maximum weight
Dotted line is best fit, i.e., RECIPROCAL MODEL
```

```
                SUMMARY OF FITTED EQUATIONS, WEIGHTED LEAST SQUARES

                                      A              B          R-SQUARE
LINEAR      : Y=A+B*X             .44101E+05    -.26920E+04       .86
QUADRATIC   : Y=A+B*X*X           .32996E+05    -.14056E+03       .74
CUBIC       : Y=A+B*X*X*X         .29241E+05    -.88362E+01       .63
RECIPROCAL  : Y=A+B/X            -.33078E+03     .14972E+06       .99
SQUARE ROOT: Y=A+B*sqrt(X)        .66175E+05    -.15767E+05       .92
LOGARITHMIC: Y=A+ln(X)            .66376E+05    -.22069E+05       .96
POWER       : Y=A*X**B            .14997E+06    -.10098E+01       .99
EXPONENTIAL: Y=A*exp(BX)          .10938E+02    -.12764E+00       .96
```

FIGURE 8.2:

PLOT OUTPUT FOR REGULATORY PROBLEM, MODIFICATION #1

Incidentally, IF statements can be used with plots to narrow the information being plotted or for some other specific purpose. As an example, consider the following line:

```
IF(FACTOR .GT. 10000) CALL PLOT(DEPTHL,'(DEPTHL)',FACTOR,'(FACTOR)',P1)
```

In this case, only those points will be plotted where the FACTOR variable exceeds 70,000.

8.3 MOUSE Automated Sensitivity: Introduction

As was mentioned in Chapter 1, traditional sensitivity analysis usually starts with a base value estimate, followed by a perturbation or change in one of the input variables (holding all other input variables at their previous values). The process is repeated for as many variables and for as great a change as is desired. When the sensitivity of parameters (i.e., temporal constants) on the final output is involved, different values of the parameters can be placed either in matrices or vectors and, using the MOUSE Pass, sensitivity runs can be made. For example, in Figure 7.1 the sensitivity of the present value, PVALUE, of the project to the value of the slope in the YIELD1 equation was studied by placing three different values of the slope in a matrix (a vector also could have been used).

The IF Statement discussed in Chapter 4, Sections 4.6 and 4.7 could also have been used in conjunction with the MOUSE Pass to study the sensitivity of the present value of the project to different assumptions of probability distributions for the input variables. For example, it was assumed that the CONTAM variable followed the discrete uniform distribution with parameters lower = 1.0 and upper = 4.0. On three MOUSE passes we could have examined the effects of three distributions by replacing the CONTAM equation with the following three equations (all three of the following distribution have the same mean, i.e., 3, but their shapes are quite different:

```
IF(IPASS .EQ. 1) CONTAM=DUNI(1.0,4.0)
IF(IPASS .EQ. 2) CONTAM=TRI(1.0,3.0,5.0)
IF(IPASS .EQ. 3) CONTAM=GAU(3.0,1.0)
```

The following Section describes yet another method for making sensitivity runs in MOUSE.

8.4 The Sensitivity Control Line

With a single control line (the Sensitivity Control Line), it is possible to alter either the mean or the standard deviation of any stochastic input variable by a stated percentage (up or down) to see what effect such a change has on the output variables. When this option is used, the following takes place:

1. First, a normal iteration sequence occurs, i.e., there is no change in the original parameters or the form of the input stochastic variables (the so-called "base run");
2. Then, for every Sensitivity Control Line in the program, another iteration sequence automatically occurs, incorporating the change in parameter specified in that line and only in that line (the so-called "sensitivity run").

It also should be noted that tables, vector tables, and plots (the last two to be discussed shortly) are clearly labeled as to whether they are "base" runs or "sensitivity" runs. Furthermore, on sensitivity runs each table contains the percent change from the base table mean and standard deviation.

The Sensitivity Control Line has the following form:

```
CALL SEN(variable,'(lname)',m%,std%)
```

where "variable" is the stochastic input variable to be altered, "lname" is a literal name (note the single quotes and parentheses) to be assigned to the variable during printout (usually this is the same as the variable name), "m%" is the desired change in the mean, "std%" is the desired change in the standard deviation. Also note that the variable name must be a decimal name. (A decimal name is one that starts with any letter except I, J, K, L, M, or N. See Chapter 3, Section 3.2). A maximum of 50 Sensitivity Control Lines may be used in any MOUSE program. One can change both the mean and the standard deviation of a variable simultaneously, but this is not recommended since it would not be possible to sort out separately the two effects. It should be mentioned that although changing the parameters of a probability distribution may alter the shape of the distribution, the underlying distribution remains the same, e.g., a normal distribution still remains a normal, a uniform distribution still remains a uniform, etc.

The following example is taken from Figure 8.1:

```
CALL SEN(RLEACH,'(RLEACH)',50.,0.)
CALL SEN(RLEACH,'(RLEACH)',0.,-50.)
```

Figure 8.1 shows the placement of these two Sensitivity Control

```
                    ********************************************
                    *    DISTRIBUTION FOR QUANTITY FACTOR      *
                    ********************************************

NUMBER OF ITERATIONS =  5000          MEAN =     20361.21000                    STANDARD DEVIATION =     12154.1600
                                   MINIMUM =      4219.64600           COEFFICIENT OF VARIATION, % =        59.6927
                                   MAXIMUM =     82200.32000


                    ********************************************
                    *    DISTRIBUTION FOR QUANTITY FACTOR      *
                    ********************************************

THIS TABLE IS A RESULT OF A SENSITIVITY TEST FOR RLEACH  :
  MEAN OF RLEACH   IS INCREASED BY  50.0 PERCENT.
  STANDARD DEVIATION OF RLEACH   REMAINS UNCHANGED.

NUMBER OF ITERATIONS =  5000          MEAN =     20737.69000                    STANDARD DEVIATION =     12264.3100
                                   MINIMUM =      4716.75000           COEFFICIENT OF VARIATION, % =        59.1402
                                   MAXIMUM =     89238.35000

CHANGE FROM BASE FACTOR   VALUES DUE TO THIS CHANGE IN RLEACH  :

           MEAN:          1.8490%
STANDARD DEVIATION:        .9063%


                    ********************************************
                    *    DISTRIBUTION FOR QUANTITY FACTOR      *
                    ********************************************

THIS TABLE IS A RESULT OF A SENSITIVITY TEST FOR RLEACH  :
  MEAN OF RLEACH   REMAINS UNCHANGED.
  STANDARD DEVIATION OF RLEACH   IS DECREASED BY  50.0 PERCENT.

NUMBER OF ITERATIONS =  5000          MEAN =     20292.22000                    STANDARD DEVIATION =     11396.3800
                                   MINIMUM =      5335.72300           COEFFICIENT OF VARIATION, % =        56.1613
                                   MAXIMUM =     83532.23000

CHANGE FROM BASE FACTOR   VALUES DUE TO THIS CHANGE IN RLEACH  :

           MEAN:          -.3388%
STANDARD DEVIATION:      -6.2347%
```

FIGURE 8.3:

SENSITIVITY OUTPUT FOR THE REGULATORY PROBLEM, MODIFICATION #1

Lines. (They can be placed, preferably, immediately after the Stochastic Variable Sampling Line that generates the variable or, in most cases, immediately after the last Stochastic Variable Sampling Line.) The first line increases the mean of the variable RLEACH by 50% and the second decreases its standard deviation by 50%. (Note: Any attempt to decrease the standard deviation by more than 100% will result in an error message and the run will be terminated.) A sample output for the first sensitivity run is shown in Figure 8.3.

Note that MOUSE does not require you to know either the mean or the standard deviation of a stochastic input variable in order to use Sensitivity Control Lines. All you have to specify is a <u>relative</u> increase or decrease in these parameters. If Sensitivity Control lines are used, the last executable statement in any MOUSE program <u>must</u> be a <u>Standard</u> Table Output line.

8.5 The Sensitivity Level Line

MOUSE output is rather prodigious. The standard MOUSE table discussed in Chapter 6 ordinarily fills one page; the Plot (see next Section) and Vector Table (see Chapter 10) outputs generally run to two pages each. If many sensitivity runs are scheduled or if you are interested only in the major effects reflected in such runs, such a level of output may not be desired. An option is provided in MOUSE whereby you may select up to four levels of output when sensitivity runs are to be made. The level desired is selected by a <u>Sensitivity Level Line</u> which is placed in the Specification Block of the program. This line has the following form:

```
SENS=c
```

where c, the sensitivity level, is one of three integers as follows:

Level	Description
0	No curtailment of plots or tables
1	All plots deleted (except for base iteration);
2	All vector tables <u>and</u> plots deleted (except for base iteration);
3	All vector tables and plots deleted; standard tables automatically converted to short form (except for base iteration).

The 3 level of sensitivity is achieved by default, i.e. by the absence of the Sensitivity Level line in the program. Note that the use of a Sensitivity Level line in no way affects the base output, i.e., the output of the first iteration sequence. Only the sensitivity runs themselves will be affected.

See Chapter 11, Section 11.6, for a discussion of TRAP, a MOUSE utility that greatly facilitates the preparation of the Sensitivity Control and Level lines. Also, CHECKER, Chapter 11, Section 11.2, will examine a MOUSE source file to insure that arguments of these lines are correct.

EXERCISES

Which of the following are valid MOUSE calls?

```
1. CALL PLOT(X1,'(PRICE)',X2,'(QUANTITY),T1)
2. CALL SEN(AREA,'(AREA)',-120.,0.)
3. CALL PLOT(X1,'(PRICE)',X2/1000,'(QUANTITY)',T1)
4. CALL SEN(AREA,'(AREA)',0.,-120.)
5. CALL PLOT(X1,'(X1)',PVALUE,'(PVALUE)',T1)
```

Which of the following program line sequences, if any, are valid?

6.

```
                    .
                    .
                    .
      CONTAM=EDIS(D1,5)
      CALL SEN(CONTAM,'(CONTAM)',0.,50.)
      PRICE=EDIS(D2,5)
      COST=EDIS(D3,5)
      CALL SEN(COST,'(COST)',0.,50.)
                    .
                    .
                    .
```

7.

```
                    .
                    .
                    .
      CONTAM =EDIS(D1,3)
      PRICE=EDIS(D2,3)
      CALL PLOT(CONTAM,'(CONTAM)',PRICE,'(PRICE)',T1)
 10   CONTINUE
      GOTO(5,15), JUMP
 15   CONTINUE
      END
```

CHAPTER 9

CORRELATED STOCHASTIC INPUT VARIABLES

9.1 Introduction

It will be noted that in the Waste Oil Reclamation Problem of Figure 2.1 (and also in the modification shown in Figure 7.1), all of the stochastic input variables are independent of each other. In many cases this is not a realistic situation since correlated variables (i.e., variables which are likely to vary together in a systematic way) are frequently encountered in practice. In times of high inflation, such have occurred in recent years, the independence of variable costs and prices is certainly questionable. Product or project life is related to market share and market growth, and price and market size are often related. Many examples from the sciences can also be cited. As Pouliquen (1970) remarks: "Correlated variables are difficult to detect, and even more difficult to measure, but overlooking them may lead to a completely wrong interpretation in the analysis".

9.2 Linear Correlation

The simplest kind of correlation between two variables is that expressed in the form of the following linear equation:

$$Y = slope*X + intercept + error \qquad [9.1]$$

Since the slope is a function of the correlation between the variables X and Y, equation 9.1 can be written as:

$$Y = (sx/sy)*rho*X + intercept + error \qquad [9.2]$$

where sx and sy are the standard deviations of variables X and Y, respectively, and rho is the correlation coefficient between X and Y. Usually, the error is assumed to be distributed normally, with mean = zero and standard deviation = std so that equation 9.2 can be written as:

$$Y = (sx/sy)*rho*X + intercept + GAU(0.0,std) \qquad [9.3]$$

As an example, suppose in the we replace the equation for the throughput in production of motor oil grade 1, THRU1=10.0-CONTAM, with the equation,

THRU1=(1.3/2.0)*0.8*CONTAM+3.0+GAU(0.0,0.5)

where 1.3 is the standard deviation of CONTAM, 2.0 is the standard deviation of THRU1, and the error term is Normal with mean = 0.0 and standard deviation = 0.5. The correlation coefficient between X and Y is 0.8.

IMP (Interactive Modeler for Probabilities) has already been introduced in chapter 4. It can be used to analyze data pairs, including fitting a variety of X-versus-Y-type equations. The user has a choice between placing the (X,Y) observation pairs into a data file (named IMP.DAT) accessible to the program or entering the data directly from the keyboard. The first line of the data file contains descriptive information (up to 60 characters) used to identify the file. (This line is mandatory.) The remainder of the file contains the sample data, one (X,Y) observation pair per line starting in column 1, with the use of the decimal point mandatory. The X and Y values are separated by either a comma or a space. Figure 9.1 shows part of such a file for a 100-observation data pair set consisting of numbers generated by the following equation:

Y = 5 + 10.*X*X + GAU(0.,20.)

where GAU(0.,20.) is a normally distributed error term and the X-values were random over the interval (1,10).

As mentioned in Chapter 4, IMP prompts the user for instructions and offers a series of menus and options as follows:

1. After calling IMP (simply by typing "IMP"), you will be asked to supply a title or comment line (not to exceed 60 characters) that will be placed in the IMP output file. If the file IMP.OUT (i.e., the standard IMP output file) exists, you will be asked whether you wish to overwrite this file or not. If not, you will asked for the name of an alternate output file.

2. The following menu is presented:

```
***********************************************
* FIT A DATA SET TO A CLASSICAL DISTRIBUTION *
* INTEGRATE A CONTINUOUS  DISTRIBUTION        *
* ANALYZE A BIVARIATE DISTRIBUTION            *
* ANALYZE A DATA SET FOR AUTOCORRELATION      *
***********************************************
```

```
BIVARIATE ANALYSIS: QUADRATIC  Y=5+10.*X*X+GAU(0.,20.)
    5.50    307.50
    2.69     99.97
    9.61    932.88
    8.42    720.17
    2.22     60.04
    8.49    692.86
    3.30     98.94
    5.68    333.54
    2.42     80.78
    6.58    443.61
    2.73     86.43
    2.58    114.06
    9.28    884.38
```

FIGURE 9.1:

PORTION OF EXAMPLE IMP DATA FILE FOR ANALYSIS OF A BIVARIATE DISTRIBUTION

The choices on the menu are highlighted. For the purpose of analyzing a bivariate data set, select the third option, "ANALYZE A BIVARIATE DISTRIBUTION". The IMP program was run using 100 data pairs generated by the quadratic model shown previously, and the results are shown in Figure 9.2. (Omitted are the bivariate normal and bivariate lognormal analyses that are part of this option. See Section 9.3.) The best equation is the quadratic model (R-square = .99), with estimated parameters A = 7.76 (actual value = 5), B = 9.97 (actual value = 10), and error standard deviation = 19.84 (actual value = 20). Since some X-observations were negative, neither the power or the exponential models could be fitted to this data set. Note that the Accumulation Line (Chapter 4, Section 4.10) can be used in a MOUSE program to accumulate data to be subjected to a bivariate analysis.

9.3 Bivariate Normal Distribution

Suppose in the Waste Oil Reclamation problem of Figure 7.1 Waste Oil Reclamation problem of Figure 7.1 that although it is known that PROD2 and PRICE1 are correlated, we do not know what the cause-and-effect relationship is. We assume, however, that PROD2 and PRICE1 are bivariate normally distributed with mean and standard deviations of (50.0,10.0) and (3.0,0.6), respectively. (These very closely approximate the discrete distributions defined in Table 2.1 if we assume that the bounds of the discrete distributions are roughly the plus or minus 2 standard deviation

```
     -----------------------------------------------------------
               SUMMARY OF FITTED REGRESSION EQUATIONS
                    NUMBER OF OBSERVATIONS = 100
     -----------------------------------------------------------

LINEAR MODEL: Y=A+B*X

  A CONSTANT =          -232.79310     B CONSTANT =             110.06930
  R-SQUARE   =              .95027     ERROR      =              63.90828

QUADRATIC MODEL: Y=A+B*X*X

  A CONSTANT =             7.76751     B CONSTANT =               9.97774
  R-SQUARE   =              .99521     ERROR      =              19.84336

CUBIC MODEL: Y=A+B*X*X*X

  A CONSTANT =            98.55668     B CONSTANT =               1.01323
  R-SQUARE   =              .97550     ERROR      =              44.85876

RECIPROCAL MODEL: Y=A+B/X

  A CONSTANT =           660.85100     B CONSTANT =           -1149.52900
  R-SQUARE   =              .51076     ERROR      =             200.45450

SQUARE ROOT MODEL: Y=A+B*sqrt(X)

  A CONSTANT =          -687.53300     B CONSTANT =             466.27110
  R-SQUARE   =              .88524     ERROR      =              97.08292

LOGARITHMIC MODEL: Y=A+B*ln(X)

  A CONSTANT =          -330.14710     B CONSTANT =             447.97770
  R-SQUARE   =              .78446     ERROR      =             133.05100

POWER MODEL, Y=A*X**B, OMITTED SINCE ONE
OR MORE X- OR Y- VALUES LESS THAN ZERO.

EXPONENTIAL MODEL, Y=A*exp(B*X), OMITTED
SINCE ONE OR MORE Y-VALUES LESS THAN ZERO.
```

FIGURE 9.2: IMP OUTPUT, BIVARIATE DISTRIBUTION, DATA OF FIGURE 9.1

points of the continuous distributions.) Since another parameter of a bivariate normal distribution is rho, the correlation coefficient, some practical way must be found to determine this value.

We start by noting that the slope, b_1, of a linear regression of y on x can also be expressed (see Draper and Smith, 1966) as:

$$b_1 = rho(sy/sx) \qquad [9.4]$$

where sy and sx are the standard deviations of y and x, respectively. Since the intercept term, b_o, is my - $b_1$mx, where my and mx are the averages of y and x, respectively, the regression equation can be written as:

$$y = my + rho(sy/sx)(x- mx) \qquad [9.5]$$

Solving for rho we obtain:

$$rho = (y - my)(sx)/(x - mx)(sy) \qquad [9.6]$$

After my, mx, sy, and sx have been determined, the value of one variable is estimated at two different values of the other, preferably a value on each side of the mean. To illustrate, suppose the analyst thinks that, if the price is 1.5 then the production will be 67.5, and that if the price is 4.5 then the production will be 27.5. These values are then used in Equation 9.3 to determine rho as follows:

$$rho = (67.5 - 50)(0.6)/(1.5 - 3)(10) = -0.7$$

$$rho = (27.5 - 50)(0.6)/(4.5 - 3)(10) = -0.9$$

Averaging these two values we obtain a value of rho of -0.8. (The procedure is due to Hillier, 1964.) Note: If data is available, the correlation coefficient can also be estimated by IMP (see Section 9.5).

The bivariate normal distribution function is called by:

```
BGAU1(mx,sx,Z)
BGAU2(my,sy,rho,Z)
```

where the m's and s's denote the means and standard deviations of the two variables, respectively, rho is the correlation coefficient, and Z is a dummy variable used to link the two distributions. It is not necessary to change the dummy variable designation of "Z" for different bivariate normal distributions, but for each bivariate normal distribution function called, no other bivariate normal distribution line can fall between the BGAU2 line and the BGAU1 line. For example, the Stochastic Variable Sampling Lines for PROD2 and PRICE1 in Figure 3.1 are now replaced with:

```
PROD2=BGAU1(50.0,10.0,Z)
PRICE1=BGAU2(3.0,0.6,-0.8,Z)
```

as shown in Figure 8.2. It does not matter which variable is chosen to be placed in the BGAU1 function.

The bivariate normal distribution can be used to chain variables in an interesting manner. Consider the following set of three lines in a MOUSE program:

```
X1=BGAU1(200.,40.,Z)
X2=BGAU2(700.,140.,0.8,Z)
X3=BGAU2(400.,80,0.6,Z)
```

Note that only the first line utilizes the BGAU1 function. The result is the production of three variables, each correlated with one another. The correlation between X1 and X2 is, as expected, 0.8; the correlation between X2 and X3 is that used in the third or X3 line, i.e., 0.6. The correlation between X1 and X3 is the product of all the correlations, i.e., 0.8(0.6) = 0.48. You can chain as many variables as is desired.

For changing means or standard deviations, Sensitivity Control Lines can also be used with correlated variables (dependency unknown) in the usual manner. Such Sensitivity Control Lines should be placed after the last BGAU2 line in the bivariate chain to which they refer. Sensitivity Control Lines can also be used with correlation coefficients but the following special procedure must be used:

a. Assign a name to the correlation coefficient in question;

b. Assign a value to this name;

c. Place a Sensitivity Control line after the assignment in (b);

d. Place these two lines immediately before the BGAU2 line in which the correlation coefficient is to appear;

e. In the BGAU2 line, use the name assigned for the correlation coefficient, rather than the original number.

For example, consider the three variables chained in the last example. Suppose we wish to, on successive sensitivity runs, increase the first correlation coefficient by 10% and the decrease the second by 30%. The lines would appear as follows:

```
X1=BGAU1(200.,40.,Z)
RHO1=0.8
CALL SEN(RHO1,'....RHO1',10.,0.)
X2=BGAU2(700.,140.,RHO1,Z)
RHO2=0.6
CALL SEN(RHO2,'....RHO2',-30.,0.)
X3=BGAU2(400.,80.,RHO2,Z)
```

Warning: Correlation coefficients are numbers between -1 and 1. Be careful not to alter the value of a correlation coefficient beyond this range or else an error message will be printed by MOUSE and your run will be terminated automatically.

9.4 Bivariate Lognormal Distribution

MOUSE provides another bivariate distribution, the lognormal. The general form of the bivariate lognormal distribution function is:

```
BGAUL1(mx,sx,X,Y,Z)
BGAUL2(my,sy,rho,X,Y,Z)
```

Except for the additional dummy variables (X and Y), the discussion of this Section is identical to that of Section 9.3. There are three slight differences:

a. The permissible range of the correlation coefficient of a bivariate lognormal distribution is:

$$-1/[1 + (sx)(sy)/(mx)(my)] \text{ to } +1$$

b. If the bivariate lognormal correlation coefficients are of the same sign, the geometric mean is used to average them, i.e.,

both positive then rho = $[(rho1)(rho2)]^{0.5}$
both negative then rho = $-[(rho1)(rho2)]^{0.5}$
different signs then rho = $(rho1 + rho2)/2$

c. The procedure for estimating the correlation coefficient must be modified as follows:

$$\text{let } mx = \ln[mx'(1 + rx)^{-0.5}] \qquad [9.7]$$
$$my = \ln[my'(1 + ry)^{-0.5}] \qquad [9.8]$$
$$sx = [\ln(1 + rx)]^{0.5} \qquad [9.9]$$
$$sy = [\ln(1 + ry)]^{0.5} \qquad [9.10]$$
$$\text{where } rx = (sx')^2/(mx')^2 \qquad [9.11]$$
$$ry = (sy')^2/(my')^2 \qquad [9.12]$$

the primes denoting the lognormal means and standard deviations. The x and y values in Equation 9.3 are modified also:

$$x = \ln x' \qquad [9.13]$$
$$y = \ln y' \qquad [9.14]$$

Equation 9.3 is used to calculate rho as before, and the lognormal correlation coefficient is obtained by:

$$rho' = [(1 + rxy)^{rho} - 1]/rxy \qquad [9.15]$$

where

$$rxy = (sx')(sy')/(mx')(my') \qquad [9.16]$$

Remember that the geometric mean is used to average bivariate lognormal correlation coefficients if they are of the same sign.

As an example, we use the data of Section 9.3 to calculate the correlation coefficient for a bivariate lognormal distribution. These data are as follows:

$mx' = 3$
$my' = 50$
$sx' = 0.6$
$sy' = 10$
1st (x',y') pair = $(1.5,67.5)$
2nd (x',y') pair = $(4.5,34.5)$

(The last value of the 2nd pair of data points has been changed to 34.5 in order to remain within the allowable limits of a bivariate lognormal distribution.)

Using equations 9.8, 9.9, and 9.13:

$$rx = 0.6^2/3^2 = .04$$
$$ry = 10^2/50^2 = .04$$
$$rxy = (0.6)(10)/(3)(50) = .04$$

Using equations 9.4, 9.5, 9.6, and 9.7:

$$mx = \ln[3(1 + .04)^{-0.5}] = 1.0790$$
$$my = \ln[50(1 + .04)^{-0.5}] = 3.8336$$
$$sx = [\ln(1 + .04)]^{0.5} = .1980$$
$$sy = [\ln(1 + .04)]^{0.5} = .1980$$

Using equations 9.10 and 9.11:

1st pair: $x = \ln 1.5 = .4055$ $y = \ln 67.5 = 4.2121$
2nd pair: $x = \ln 4.5 = 1.5041$ $y = \ln 34.5 = 3.5410$

Using the first pair of x/y values in equation 9.3 we obtain:

$$\text{rho} = (4.2121 - 3.8336).1980/(.4055 - 1.0790).1980 = -.5620$$

Using equation 9.12:

$$\text{rho}' = [(1 + .04)^{-0.5620} - 1]/.04 = -0.5450$$

Using the second pair of x/y values in equation 9.3 we obtain:

$$\text{rho} = (3.5410 - 3.8336).1980/(1.5041 - 1.0790).1980 = -0.6883$$

Using equation 9.12:

$$\text{rho}' = [(1 + .04)^{-0.6883} - 1]/.04 = -0.6659$$

Averaging the values of rho' of -0.5450 and -0.6659 we obtain the final value of rho':

$$\text{rho}' = -[(.5450)(.6659)]^{0.5} = -0.6024.$$

Note: If data is available, the uncorrected correlation coefficient, rho, can also be estimated by IMP (see Section 9.5).

9.5 Using IMP to Analyze Bivariate Normal And Bivariate Lognormal Distributions

IMP can be used to analyze data pairs for bivariate normal and bivariate lognormal distributions. The user has a choice between placing the (X,Y) observation pairs into the data file IMP.DAT or entering the data directly from the keyboard. The first line of the data file contains descriptive information (up to 60 characters) used to identify the file. (This line is mandatory.) The remainder of the file contains the sample data, one (X,Y) observation pair per line starting in column 1, with the use of the decimal point mandatory. The X and Y values are separated by either a comma or a space. Figure 9.3 shows part of such a file for a 100-observation data pair set consisting of numbers generated by the following pair of MOUSE calls:

```
X1=BGAU1(100.,25.,Z)
X2=BGAU2(50.,10.,0.8,Z)
```

```
BIVARIATE NORMAL: X1=BGAU1(100.,25.,Z), X2=BGAU2(50.,10.,0.8,Z)
  100.00      1.00
   89.48     45.93
  106.23     49.66
  107.13     58.35
   83.86     40.39
  112.08     54.78
  121.52     57.94
   84.66     47.18
  131.95     60.41
  123.32     59.19
   87.83     47.37
   81.05     44.93
```

FIGURE 9.3:

PORTION OF EXAMPLE IMP DATA FILE FOR ANALYSIS OF A BIVARIATE NORMAL DISTRIBUTION

IMP prompts the user for instructions and offers the series of menus and options shown previously.

1. After calling IMP (simply by typing "IMP"), you will be asked to supply a title or comment line (not to exceed 60 characters) that will be placed in the IMP output file. If the file IMP.OUT (i.e., the standard IMP output file) exists, you will be asked whether you wish to overwrite this file or not. If not, you will asked for the name of an alternate output file.

2. The following menu is presented:

```
**********************************************
* FIT A DATA SET TO A CLASSICAL DISTRIBUTION *
* INTEGRATE A CONTINUOUS  DISTRIBUTION       *
* ANALYZE A BIVARIATE DISTRIBUTION           *
* ANALYZE A DATA SET FOR AUTOCORRELATION     *
**********************************************
```

For the purpose analyzing a data set for bivariate normal or bivariate lognormal distributions, select the third option, "ANALYZE A BIVARIATE DISTRIBUTION". The IMP program was run using 100 data pairs generated by the bivariate normal model shown above, and the results are shown in Figure 9.4. (Omitted are the regression fits that are part of this option. See Section 9.2.) Note that the mean and standard deviation parameters for variables X1 and X2 have been estimated as (100.0,24.2) and (49.2,10.4), respectively (the actual values were (100,25) and

```
---------------------------------------------------------------
               BIVARIATE DISTRIBUTION ANALYSIS
 (Numbers in parentheses are significance levels of coefficients)
---------------------------------------------------------------


CORRELATION COEFFICIENT =  .6990

STATISTICS FOR 1st VARIABLE:
                 MEAN =                100.0303
  STANDARD DEVIATION =                  24.2179
  COEFFICIENT OF SKEWNESS (UNTRANSFORMED) =       .03 (.458)
  COEFFICIENT OF KURTOSIS (UNTRANSFORMED) =       .43 (.183)
  COEFFICIENT OF SKEWNESS (LOG TRANSFORM) =     -1.20 (.999)
  COEFFICIENT OF KURTOSIS (LOG TRANSFORM) =      3.77 (.001)


STATISTICS FOR 2nd VARIABLE:
                 MEAN =                 49.2156
  STANDARD DEVIATION =                  10.4783
  COEFFICIENT OF SKEWNESS (UNTRANSFORMED) =      -.94 (.999)
  COEFFICIENT OF KURTOSIS (UNTRANSFORMED) =      3.32 (.001)
  COEFFICIENT OF SKEWNESS (LOG TRANSFORM) =     -7.14 (.999)
  COEFFICIENT OF KURTOSIS (LOG TRANSFORM) =     62.88 (.001)

  Note:
  *  = .01 < p and p < or = .05
  ** =  p < or = .01
```

FIGURE 9.4:

IMP OUTPUT, BIVARIATE NORMAL DISTRIBUTION ANALYSIS, DATA OF FIGURE 9.3

(50,10), and the correlation coefficient was estimated as 0.7 (the actual value was 0.8).

9.6 Time Series and Autocorrelation

If a stochastic process produces random variables such that there is a random variable x_t for each value of t (where t indicates time), we observe a random function of time called a <u>Time Series</u>. A common property of many time series related to economic and technological phenomena is the relationship of x_{t+j} and x_t, where j is the lag, i.e., the number of time intervals between the respective values of the time series. If x_{t+j} and x_t are independent, then their covariance, cov(x_t,x_{t+j}), will be zero. If this covariance is not zero, then the time series is said to be Autocorrelated.

Suppose we define an Autocorrelation Function as follows:

$$rho(j)=cov(x_t,x_{t+j})/cov(x_t,x_{t+0}) \qquad [9.17]$$

where, since j=0 implies no lag, the cov(x_t,x_{t+0}) term = var(x_t). We also assume that the expected or mean value of x_t is zero. These conditions are valid for stationary time series.

It is not generally possible to generate a time series with an arbitrary autocorrelation function. There are two special functions, however, which can be used with satisfactory results and flexibility if the distribution of the x_t variables is normal.

(a) Linear Autocorrelated Function

Let

$$\begin{aligned} rho(j) &= 1 - j/m && \text{for } j < m && [9.18] \\ &= 0 && \text{for } j >= m \end{aligned}$$

This is a linearly decreasing function of j and a model for autocorrelated time series where zero autocorrelation can be assumed for lags equal to or larger than m. A linearly autocorrelated time series of normally distributed variables, x, can be generated by the following MOUSE line:

```
CALL AUTOL(mean,std,k,value)
```

where "mean" and "std" are decimal variables representing the mean and standard deviation of the variable, respectively, "k" is an integer variable equal to 12/m, and "value" is the name of the variable. AUTOL uses a fast approximation to the normal distribution involving the generation of 12 uniform variables, thus the k = 12/m identity and subsequent limitation of this subroutine to 12 lags. Because of the limits on the ranges of m in equation 9.18, AUTOL is limited to a maximum of 12 lags. Consequently, the practical values of k are as follows:

k	# of lags
1	12
2	6
3	4
4	3
6	2

A typical call, therefore, would look like:

```
CALL AUTOL(100.,25.,2,PRICE)
```

PRICE, therefore, would be generated as a normally distributed, linearly autocorrelated time series (up to 6 lags) with mean equal to 100 and standard deviation equal to 25.

(b) Exponential Autocorrelated Function

In an exponential autocorrelated function, the coefficient of correlation with lag j is expressed as

$$\mathrm{rho}(j) = \mathrm{lambda}^k \qquad 0 < \mathrm{lambda} < 1 \qquad [9.19]$$

It can be shown that this type of autocorrelation function exists in time series obtained by exponential smoothing based on the recursion relation,

$$x_o = (1 - \mathrm{lambda})r_o$$

$$x_t = \mathrm{lambda}x_{t-1} + (1 - \mathrm{lambda})r_t \qquad [9.20]$$

where the r_t numbers are mutually independent variables with zero mean and variance sigma.

An exponential autocorrelated time series of normally distributed variables, x, can be generated by the following MOUSE line:

```
CALL AUTOE(mean,std,lambda,value)
```

where "mean" and "std" are decimal variables representing the mean and standard deviation of the variable, respectively. A typical call, therefore, would look like:

```
CALL AUTOE(100.,25.,0.8,PRICE)
```

PRICE would be generated as a normally distributed, exponential autocorrelated time series with lambda equal to 0.8, mean equal to 100, and standard deviation equal to 25.

IMP can be used to analyze a data set for autocorrelation (Option four in the Main Menu). The user has a choice between placing the observations into the usual data file IMP.DAT or entering the data directly from the keyboard. The first line of the data file contains descriptive information (up to 60 characters) used to identify the file. (This line is mandatory.) The remainder of the file contains the sample data, one observation per line starting in column 1, with the use of the decimal point mandatory. Figure 9.5 shows part of such a file for a 100-observation data pair set consisting of numbers generated by the following MOUSE call:

```
CALL AUTOE(100.,25,,0.8,X)
```

Partial results are shown in Figure 9.6. Note the exponentially decreasing autocorrelation coefficient as the lag increases. Note that the Accumulation Line (Chapter 4, Section 4.10) can be used in a MOUSE program to accumulate data to be subjected to an autocorrelation analysis.

EXERCISES

Which of the following are valid statements or sequences?

1.
```
Y6=BGAUL1(200.,20.,X,Y,Z)
Y7=BGAUL2(100.,10.,0.7,X,Y,Z)
```

2.
```
X1=BGAU1(50.,10.,Z1)
X2=BGAU2(100.,55.,0.8,Z2)
```

3.
```
X1=BGAU1(200.,40.,Z)
X2=BGAU2(700.,140.,0.8,Z)
X3=BGAU2(400.,80.,-0.6,Z)
X4=BGAU2(100.,20.,0.8,Z)
```

5. Given the following data, find the bivariate lognormal correlation coefficient:

mx = 100, my = 200, sx = 10, sy = 40
when x = 80, y = 150 and when x = 120, y = 250

REFERENCES

Draper, N.R., and Smith, H., Applied Regression Analysis, John Wiley & sons, Inc., New York, N.Y., p. 35 (1966).

Hillier, F.S., The Evaluation of Risky Interrelated Investments, Technical Report No. 73, Contract Nonr-225(53) (NR-642-002) with the Office of Naval Research (1964).

Pouliquen, L. Y., Risk Analysis in Project Appraisal, World Bank Staff Occasional Papers No. Eleven, The Johns Hopkins University Press, Baltimore, p. 1-95 (1970).

```
AUTOCORRELATION: AUTOE(100.,25.,0.8,X)
100.00
 83.75
104.19
 97.69
121.86
125.43
141.76
150.28
161.361
135.90
109.79
112.41
```

FIGURE 9.5: PORTION OF EXAMPLE IMP DATA FILE FOR ANALYSIS OF AN AUTOCORRELATION (EXPONENTIAL) TIME SERIES

```
            AUTOCORRELATION ANALYSIS
    SUMMARY OF AUTOCORRELATION COEFFICIENTS:

            AUTOCORRELATION
     LAG       COEFFICIENT

      0        1.0000
      1         .7694   (.001)**
      2         .6234   (.001)**
      3         .5129   (.001)**
      4         .4013   (.002)**
      5         .3305   (.009)**
      6         .2504   (.040)*
      7         .2254   (.060)
      8         .1382   (.173)
      9         .0228   (.439)
     10        -.0419   (.610)
     11        -.0961   (.738)
     12        -.0745   (.689)
     13        -.0471   (.621)
     14         .0161   (.458)
     15        -.0001   (.500)
     16        -.0514   (.630)
     17        -.0389   (.598)
     18        -.0679   (.668)
     19        -.0878   (.711)
     20        -.0779   (.689)

     Note:  *  = .01 < p and p < or = .05
 ** =  p < or = .01
```

FIGURE 9.6: IMP OUTPUT, AUTOCORRELATED (EXPONENTIAL TIME SERIES, DATA OF FIGURE 9.5

CHAPTER 10

NON-UNIFORM CASH FLOWS, VECTORS, AND THE INTERNAL-RATE-OF-RETURN FUNCTION

10.1 Introduction

With the exception of the single payment interest functions, SCA, SPW, and CSPW, all of the interest functions described in Chapter 5 assume either a constant cash flow for each time period or else a constant change in cash flow (functions of the latter type include UPWP, GCA, and GPW). The nature of these interest functions makes it very easy to deal with a stochastic project life in MOUSE as project life is merely one of the calling arguments; it makes no difference to MOUSE whether or not these arguments are stochastic variables or constants. For example, in Figure 3.2 the annual net benefit, ANETB, is the same for all years, i.e.,

```
ANETB=GREV1+GREV2+GREV3-70.0
```

so that the present value can be calculated by use of one of the uniform series interest formulas, UPW, i.e.,

```
PVALUE=ANETB*UPW(0.08,XLIFE,0.0)-XININV
```

However, suppose the annual net benefits are increased by 12% each year (this is one of the assumptions made in modification #2 to the Waste Oil Reclamation Project problem shown in Figure 10.1), i.e.,

```
ANETB(LOOP)=SCA(.12,YEAR,0.)*(GREV1+GREV2+GREV3-70.)
```

using the SCA function for the increase and noting that YEAR and LOOP are identical (going from 1 to XLIFE) except that LOOP is an integer representation in MOUSE while YEAR represents a decimal number. Clearly ANETB is now a vector (the index for a vector in MOUSE is <u>always</u> LOOP) and UPW cannot be used. Each element of ANETB could be brought back to its present value by the SPW function; following this, all of the elements in ANETB would then have to be summed.

```
C
C WASTE OIL RECLAMATION PROBLEM MODIFICATION #2
C
      DIMENSION D1(3,2),D2(4,2),D3(4,2)
C
C VECTOR DIMENSION LINE
C
      DIMENSION ANBEN(15)
C
C VECTOR TABLE DIMENSION LINE
C
      DIMENSION VTAB(7,15)
C
      DIMENSION TAB(46)
      COMMON/Q1/C80/Q2/JUMP/Q3/LOOP/Q4/SENS/Q5/IDATA/Q6/IOUT/Q7/IPASS/Q8
     &/IRAN/Q9/IREAD/Q10/ISCR1/Q11/ISCR2/Q12/ISNAP/Q13/ITER/Q14/ITOG/Q15
     &/IWRIT/Q16/IXXX/Q17/OK1/Q18/OK2/Q19/OK3/Q20/OK4/Q21/OK5/Q22/OK6
      CALL NAME('(WASTE OIL RECLAMATION PROBLEM - MODIFICATION #2)')
      CALL READ(D1,3)
      CALL READ(D2,4)
      CALL READ(D3,4)
      ITER=5000
5     DO 10 IXXX=1,ITER
      CONTAM=DUNI(1.0,4.0)
      RAND1=CUNI(-10.0,10.0)
      PRICE1=EDIS(D1,3)
      SLOPE=ECTS(D2,4)
      PROD2=STEP(D3,4)
      RAND2=GAU(0.00,0.10)
      PRICE3=EBETA(6.5,8.0,9.5)
      PROD3=EXPON(18.0)
      XLIFE=DTRI(5.0,10.0,15.0)
      XININV=TRA(1600.0,2000.0,2200.0,2400.0)
      THRU1=10.0-CONTAM
      YIELD1=10.0-2.0*CONTAM
      PROD1=THRU1*YIELD1+RAND1
      GREV1=PRICE1*PROD1
      PRICE2=10.0-SLOPE*PROD2+RAND2
      GREV2=PRICE2*PROD2
      GREV3=PRICE3*PROD3
      LIFE=XLIFE
C
C VECTOR LOOP CONTROL LINE #1
C
      DO 100 LOOP=1,LIFE
      YEAR=LOOP
      ANBEN(LOOP)=SCA(.12,YEAR,0.)*(GREV1+GREV2+GREV3-70.)
C
C VECTOR TABLE DEFINITION LINE
C
      CALL VTABLE(ANBEN,'(ANBEN)',LIFE,VTAB)
C
C VECTOR LOOP CONTROL LINES #2 AND #3
C
100   CONTINUE
      LOOP=1
C
C PRESENT WORTH SUM FUNCTION
C
      PVALUE=PWSUM(ANBEN,0.08,LIFE,XININV)
C
      CALL TABLE(PVALUE,'(PRES VAL)',DUMO,DUM1,40,TAB)
10    CONTINUE
      GOTO(5,15), JUMP
15    CONTINUE
      CALL LIST('(WO2.FOR)')
      END
```

FIGURE 10.1:

MOUSE PROGRAM FOR THE WASTE OIL RECLAMATION PROBLEM, MODIFICATION #2

10.2 The Present Worth Sum Function

A function that does all of this in a single line is the MOUSE Present Worth Sum Function, PWSUM. This function is called by:

```
PWSUM(vector,i,upper,inv)
```

where "vector" is an m-dimensional vector, i is the interest rate, "upper" (an integer or integer variable) is the upper index of the vector loop, and "inv" is the initial investment. Also note that the vector name must be a decimal name. (A decimal name is one that starts with any letter except I, J, K, L, M, or N. See Chapter 3, Section 3.2). The PWSUM function cannot be used within vector loops (see Section 10.3 for an explanation of the vector loop). As with all MOUSE interest related functions, "i" and/or "upper" can be stochastic variables. Thus, as is shown in Figure 10.1, the present value of the annual benefits when these cash flows are non-uniform over time is calculated in MOUSE simply as:

```
PVALUE=PWSUM(ANBEN,.08,LIFE)-XININV
```

(In order to distinguish between a vector and a scalar quantity, ANETB has been renamed to ANBEN in Figure 10.1.) The vector ANBEN must be dimensioned using a Vector Dimension Line which has the following general form:

```
DIMENSION name1(m1),name2(m2),name3(m3),etc.
```

where "namei" is the name of the ith vector and "mi" is its dimension. (See Chapter 11, Section 11.6, for a discussion of TRAP, a MOUSE utility that greatly facilitates the preparation of the Dimension Block.) Also note that the variable name must be a decimal name. (A decimal name is one that starts with any letter except I, J, K, L, M, or N. See Chapter 3, Section 3.2). In Figure 10.1 this line appears as:

```
DIMENSION ANBEN(15)
```

since, in Modification #2 to the Waste Oil Reclamation Project problem, the maximum life of the facility has been stated to be 15 years. Note, however, that when MOUSE encounters the PWSUM line in the model it sums ANBEN only from the first year to LIFE, LIFE (being stochastic in this problem) varying from iteration to iteration.

Another feature of this function is that the user can specify the use of continuous compounding simply by entering the interest rate as a negative number, e.g.,

```
PVALUE=PWSUM(ANBEN,-.08,LIFE)-XININV
```

When MOUSE sees the negative interest rate, it automatically assumes that the user wants the interest calculated on a continuous basis (and, of course, uses the absolute value of the interest rate in its calculations). The true interest rate, however, must be used.

Finally, when "i" in the PWSUM function is set to zero, the PWSUM function serves as a convenient method for the simple summation of the terms in a vector. CHECKER, Chapter 11, Section 11.2, will examine a MOUSE source file to insure that arguments of the PWSUM function are correct, and that the dimension of the vectors in the function agree with those specified in the Dimension Block.

10.3 Calculating the Elements of a Vector

There remains the question of how the individual terms in a vector are calculated. If the equation for the calculation of these terms does not change from time period to time period (save for the numbers themselves), this calculation is simplified greatly by means of a vector loop. This loop is similar to that used in the MOUSE pass (see Chapter 7) but one more control line is involved. The loop need only surround those equations in the model which are affected by time. (This will be more fully explained later.) In Modification #2 to the Waste Oil Reclamation Project problem there is only one such equation in the model, i.e.,

```
ANBEN(LOOP)=SCA(.12,YEAR,0.)*(GREV1+GREV2+GREV3-70.)
```

The vector loop consists of three vector loop control lines, one preceding the equation(s) affected by time and two immediately afterwards. These control lines are of the following form:

```
      DO 100 LOOP=1,upper
          .
          .
          .
100   CONTINUE
      LOOP=1
```

where "upper" is an integer or integer variable representing the number of elements or coordinates to be filled in the vector. Note that this can be a deterministic number (such as 3) or a stochastic variable (such as LIFE). For Modification #2 to the Waste Oil Reclamation Project problem, the full vector loop is as follows:

```
      LIFE=XLIFE
      DO 100 LOOP=1,LIFE
      YEAR=LOOP
      ANBEN(LOOP)=SCA(.12,YEAR,0.)*(GREV1+GREV2+GREV3-70.)
100   CONTINUE
      LOOP=1
```

Since the upper limit of the loop is the life of the facility, which is a stochastic variable in this problem, and since the upper limit must be either an integer or an integer variable, it is necessary to define a new integer variable by setting LIFE=XLIFE (see Chapter 3, Section 3.2) prior to the vector loop. Furthermore, since in this problem we also wish to use the index LOOP as an argument in the SCA function and since the SCA function takes decimal variables only, it is necessary to define a new decimal variable by setting YEAR=LOOP. (Note that neither XLIFE or LOOP are altered by these statements; all that is being done is that two new variables are being created.)

Note that since XLIFE is used in the Irrigation Project problem only to compute the annual net benefits, the LIFE=XLIFE line in the program of Figure 10.1 could have been omitted by defining the variable on the left-hand side of the Stochastic Variable Sampling line <u>directly</u> as an integer, i.e.,

```
LIFE=DTRI(5.0,10.0,15.0)
```

10.4 The Use of the IF Statement in Vector Loops

In Chapter 4, Sections 4.6 and 4.7, the IF statement was introduced in conjunction with truncated and composite probability distributions. The IF statement also finds application within a vector loop when special conditions are imposed upon some coordinates and not upon others. For example, assume for Modification #2 to the Waste Oil Reclamation Project problem that there is a fixed salvage value of $5000 at the end of the project life. This condition is very easily met by the following addition placed immediately after the existing ANBEN(LOOP) line in the vector loop:

```
IF(LOOP .EQ. LIFE) ANBEN(LOOP)=ANBEN(LOOP)+5000.
```

The use of this IF statement insures that the $5000 will be added only to the last coordinate of the ANBEN vector. There are many models in which conditional calculations, such as this one, are involved. The IF statement can be very helpful in these instances.

10.5 Vector Loops and Order of Statements

Consider the following three examples:

```
Example 1:        X=CUNI(20.,30.)
                  DO 100 LOOP=1,3
                  YEAR=LOOP
                  VEC(LOOP)=QUAN*SCA(X,YEAR,0.)
             100  CONTINUE
                  LOOP=1

EXAMPLE 2:        DO 100 LOOP=1,3
                  X=CUNI(20.,30.)
                  YEAR=LOOP
                  VEC(LOOP)=QUAN*SCA(X,YEAR,0.)
             100  CONTINUE
                  LOOP=1
EXAMPLE 3:        DO LOOP=1,LIFE
                  YEAR=LOOP
                  VEC(LOOP)=QUAN*SCA(CUNI(20.,30.),YEAR,0.)
             100  CONTINUE
                  LOOP=1
```

Examples 1 and 2 produce quite different results. In Example 1, for every time period the same value of X is used as the percentage increase argument in the SCA function in the vector VEC. In Example 2, the value of X changes from time period to time period. Which is correct? The answer to this depends upon your model. The model of Example 1 assumes that the percentage increase in the quantity QUAN does <u>not</u> vary from year to year; in Example 2 the model assumes that it does.

Finally, Example 3 produces the same results as Example 2. Example 3 is an illustration of the <u>direct</u> method of introducing stochastic variables in MOUSE. (See Chapter 3, Section 3.4.) Some users may find the direct method useful as it eliminates a line of code in a program: however, it does complicate the line in which it is used.

10.6 The Vector Table

Although Sensitivity Control and Table and Plot Output lines can be used within a vector loop, only under special circumstances can the last two be used with vectors. However, a special kind of table, called a "Vector Table", has been provided for this purpose. The <u>Vector</u> <u>Table</u> <u>Output</u> <u>Line</u> has the form:

```
CALL VTABLE(variable,'(lname)',upper,ti)
```

```
               .
               .
C
C VECTOR DIMENSION LINE
C
      DIMENSION ANBEN(15)
                  -
               .
               .
C
C VECTOR TABLE DIMENSION LINE
C
      PRECISION T2(7,15)
               .
               .
               .
C
C LOOP FOR ANNUAL NET BENEFIT VECTOR
C
      LIFE=XLIFE
        --
      DO 100 LOOP=1,LIFE
      YEAR=LOOP      --
C
      ANBEN(LOOP)=SCA(.12,YEAR,0.)*(GREV1+GREV2+GREV3-70.)
        -
      CALL VTABLE(ANBEN,'(ANBEN)',LIFE,T2)
                    -              --
C
C END OF LOOP FOR ANNUAL NET BENEFIT VECTOR
C
100   CONTINUE
      LOOP=1
C
      PVALUE=PWSUM(ANBEN,.08,LIFE,XININV)
                     -         --
               .
               .
               .
```

```
Key:
     - = Vector Name
     -- = Index for Vector Loop, Integer or Integer Variable
          (used in Vector Loop Control Line 1 and PWSUM function)
```

FIGURE 10.2: EXCERPT FROM WASTE OIL RECLAMATION PROBLEM, MODIFICATION #2, SHOWING LINES RELEVANT TO VECTORS, VECTOR TABLES, AND PRESENT VALUE SUM FUNCTION.

```
                    **********************************
                    *  SUMMARY FOR VECTOR ANBEN      *
                    **********************************

                                 STANDARD   COEF. OF                                          COEF. OF      COEF. OF
PERIOD  COUNT          MEAN     DEVIATION  VARIATION        MINIMUM         MAXIMUM          SKEWNESS      KURTOSIS

   1     5000      651.05860    311.58850     47.86      -128.76200      2140.54400              .38           .16
   2     5000      729.18620    348.97740     47.86      -144.21350      2397.40900              .38           .16
   3     5000      816.68740    390.85830     47.86      -161.51910      2685.09800              .38           .16
   4     5000      914.69040    437.76010     47.86      -180.90140      3007.31000              .38           .16
   5     5000     1024.45300    490.29250     47.86      -202.60960      3368.18700              .38           .16
   6     5000     1147.38800    549.12430     47.86      -226.92270      3772.37000              .38           .16
   7     4811     1288.70400    615.96390     47.80      -254.15340      4225.05400              .37           .15
   8     4419     1442.26800    690.56230     47.88      -284.65190      4732.06100              .37           .17
   9     3765     1614.51100    766.17130     47.46      -318.81010      5299.90800              .34           .17
  10     2962     1804.77200    852.72930     47.25      -354.06310      5935.89600              .37           .22
  11     1982     2037.24500    950.96720     46.68      -396.55070      5900.91400              .35           .08
  12     1141     2261.82700   1058.32700     46.79      -436.88860      5935.98300              .39           .11
  13      571     2509.24500   1172.23800     46.72      -376.88600      6615.00100              .37           .06
  14      185     2699.94800   1269.92400     47.04       168.61490      7408.80100              .41          -.03
```

FIGURE 10.3: VECTOR OUTPUT FOR WASTE OIL RECLAMATION PROBLEM, MODIFICATION #2

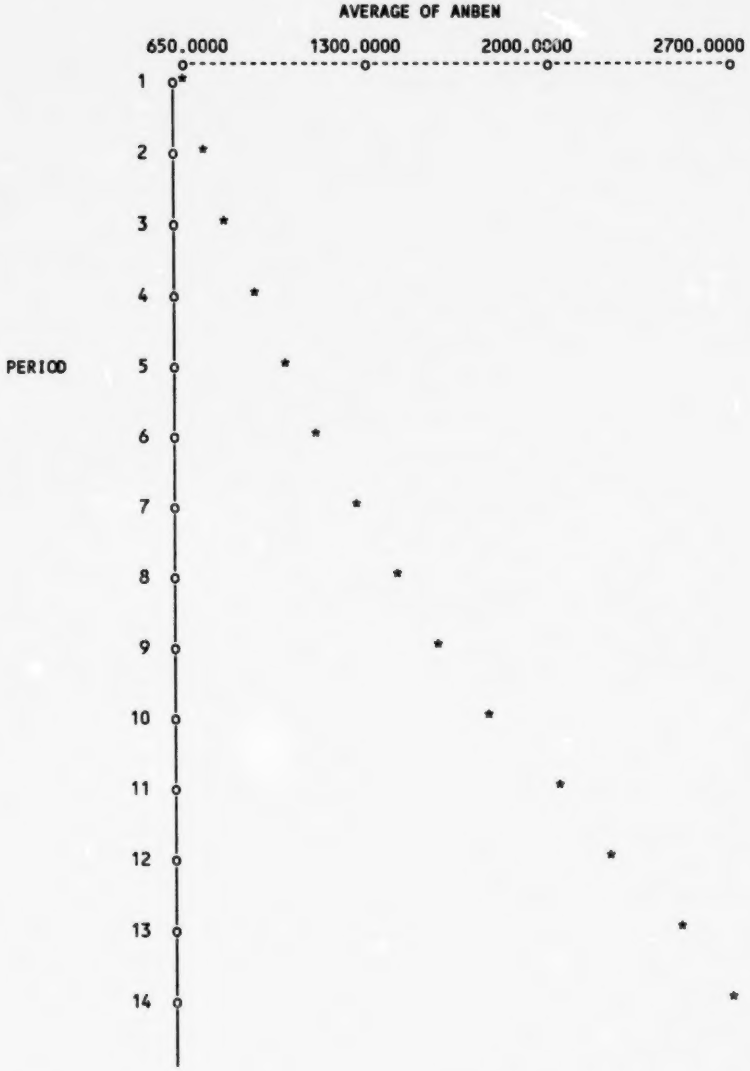


FIGURE 10.3 Continued:

VECTOR OUTPUT FOR WASTE OIL RECLAMATION PROBLEM, MODIFICATION #2

where "variable" is the vector to be tabulated, "'lname'" is an alphanumeric name for the table (note that this is a literal name and the single quotes and the parentheses must accompany the name exactly in the order shown), "upper" is the same as the "upper" specification in the vector loop (i.e., either an integer or an integer variable), and ti is the usual dummy argument used for storing the table between iterations of the program. Note that the variable name must be a decimal name. (A decimal name is one that starts with any letter except I, J, K, L, M, or N. See Chapter 3, Section 3.2). Also note that the variable can be scaled as in regular MOUSE tables (see Chapter 6, Section 6.6). In Figure 10.1 the Vector Table Output line appears as

```
CALL VTABLE(ANBEN,'...ANBEN',LIFE,T2)
```

since 15 is the dimension of the ANBEN vector and LIFE is the upper limit of the vector loop. Note that scaling (see Chapter 6, Section 6.6) is permitted in Vector Table Output lines.

As with all tables in MOUSE, the Vector Table must be dimensioned. The Vector Table Dimension Line is of the form:

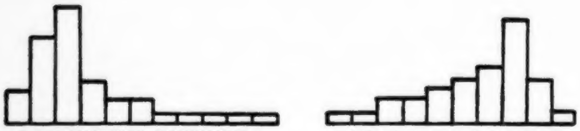
```
PRECISION T1(7,n1),T2(7,n2),T3(7,n3),etc.
```

(See Chapter 11, Section 11.6, for a discussion of TRAP, a MOUSE utility that greatly facilitates the preparation of the Dimension Block.) Also note that the variable names T1, T2, etc., must be decimal names. (A decimal name is one that starts with any letter except I, J, K, L, M, or N. See Chapter 3, Section 3.2). For example, in Figure 10.1 this line is

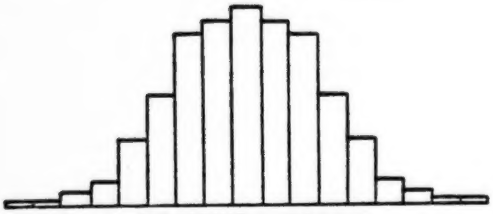
```
PRECISION T2(7,15)
```

Decimal points are, as in all dimension lines, not used. Figure 10.2 shows how the various vector-oriented lines interrelate.

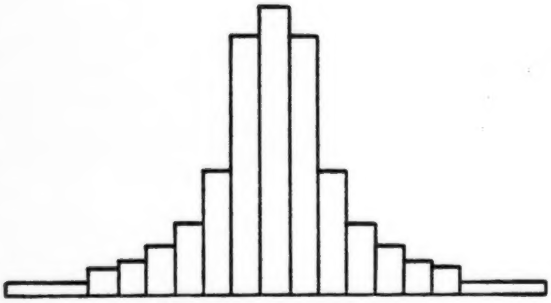
The Vector Table produced by the program of Figure 10.1 is shown in Figure 10.3. The usual table statistics of mean, standard deviation, coefficient of variation, minimum, and maximum are produced for each time period as well as two new statistics, the coefficients of skewness and kurtosis. (If, within a coordinate, the values do not vary or if the number of observations in any coordinate is less than 4, only the mean will be reported, however.) The "COUNT" in the Vector Table is equal to ITER only for those vector coordinates filled at every iteration. (On only 1613 iterations in Figure 10.3, for example, were coordinates 11 through 15 filled since there was only a one-third chance that the life of the Waste Oil Reclamation Project would reach these years.)



POSITIVELY SKEWED **NEGATIVELY SKEWED**



NEGATIVE KURTOSIS



POSITIVE KURTOSIS

FIGURE 10.4: EXAMPLES OF SKEWNESS AND KURTOSIS

For a normal distribution, the coefficient of skewness is equal to zero. A negative number indicates that the distribution is skewed to the left, i.e., that it has a long tail to the left; a positive number indicates that it is skewed to the right, i.e., a long tail to the right (see Figure 10.4). The coefficient of kurtosis for a normal distribution is also zero. A negative value indicates that the distribution is flat-topped, and a positive value indicates that the distribution has a pronounced peak (see Figure 10.4).

Finally, as an added utility, following the Vector Table is a plot of the average value of the vector versus the time period. In Figure 10.3, the sharply rising curve reflects the 8% yearly increase in net benefits programmed into Modification #2 to the Waste Oil Reclamation problem. CHECKER, Chapter 11, Section 11.2, will examine a MOUSE source file to insure that arguments in the Vector Table Definition line are correct, and that the dimensions of the dummy argument used for storing the table between iterations agree with those specified in the Dimension Block.

10.7 Vectors, The MOUSE Pass, And Plots

In Figure 7.1, the Waste Oil Reclamation Problem, Modification #1, used a matrix to store three different values of the coefficient (COEF) in the YIELD1 equation. This program produced three separate tables, one for each value of COEF. An entirely different approach has been taken in Figure 10.5 (Waste Oil Reclamation Problem, Modification #3) where the standard table has been replaced with a vector table. COEF is now a vector with index LOOP instead of IPASS; the Pass Control cards are no longer needed. Also, rather than bother reading the values of this vector, they are specified directly in the program. Finally, PVALUE has been dimensioned as a three-element vector. Since tables are not used in this program, a Dummy Table (with argument = 1, i.e., output requested) is the last executable statement in the program.

The vector table produced is shown in Figure 10.6. This is a concise summary of the effects on present value (PVALUE) of varying this coefficient. Indeed, the standard vector table plot below the table is tantamount to a plot of coefficient versus present value since the coefficients were selected at equal intervals. Although in general the standard table provides more information than does the vector table, the fact that present value at any coefficient is approximately normally distributed (as indicated by the coefficients of skewness and kurtosis, and their significant levels, shown in the vector table) negates most of the advantage of the standard table in this problem. Further-

```
C
C WASTE OIL RECLAMATION PROBLEM MODIFICATION #3
C
      DIMENSION D1(3,2),D2(4,2),D3(4,2)
C
C VECTOR TABLE DIMENSION LINE
C
      DIMENSION PVEC(7,3)
C
C VECTOR DIMENSION LINE
C
      DIMENSION COEF(3),PVALUE(3)
C
C PLOT DIMENSION LINE
C
      DIMENSION P1(100,3)
C
      COMMON/Q1/C80/Q2/JUMP/Q3/LOOP/Q4/SENS/Q5/IDATA/Q6/IOUT/Q7/IPASS/Q8
     &/IRAN/Q9/IREAD/Q10/ISCR1/Q11/ISCR2/Q12/ISNAP/Q13/ITER/Q14/ITOG/Q15
     &/IWRIT/Q16/IXXX/Q17/OK1/Q18/OK2/Q19/OK3/Q20/OK4/Q21/OK5/Q22/OK6
C
C NAME LINE
C
      CALL NAME('(WASTE OIL RECLAMATION PROBLEM - MODIFICATION #3)')
C
      CALL READ(D1,3)
      CALL READ(D2,4)
      CALL READ(D3,4)
C
C SPECIFICATION OF VECTOR VALUES
C
      COEF(1)=1.
      COEF(2)=2.
      COEF(3)=3.
C
C ITERATION NUMBER LINE
C
      ITER=5000
C
5     DO 10 IXXX=1,ITER
C
      CONTAM=DUNI(1.0,4.0)
      RAND1=CUNI(-10.0,10.0)
      PRICE1=EDIS(D1,3)
      SLOPE=ECTS(D2,4)
      PROD2=STEP(D3,4)
      RAND2=GAU(0.00,0.10)
      PRICE3=EBETA(6.5,8.0,9.5)
      PROD3=EXPON(18.0)
      XLIFE=DTRI(5.0,10.0,15.0)
      XININV=TRA(1600.0,2000.0,2200.0,2400.0)
```

```
C
C VECTOR LOOP CONTROL LINE 1
C
      DO 100 LOOP=1,3
C
      THRU1=10.0-CONTAM
      YIELD1=10.0-COEF(LOOP)*CONTAM
      PROD1=THRU1*YIELD1+RAND1
      GREV1=PRICE1*PROD1
      PRICE2=10.0-SLOPE*PROD2+RAND2
      GREV2=PRICE2*PROD2
      GREV3=PRICE3*PROD3
      ANETB=GREV1+GREV2+GREV3-70.0
      PVALUE(LOOP)=ANETB*UPW(0.08,XLIFE,0.0)-XININV
C
      CALL VTABLE(PVALUE,'(PVALUE)',3,PVEC)
C
C VECTOR LOOP CONTROL LINES 2 AND 3
C
100   CONTINUE
      LOOP=1
C
C PLOT DEFINITION LINE
C
      CALL PLOT(PVALUE(1),'(PVALUE)',PRICE2,'(PRICE 2)',P1)
C
C DUMMY TABLE
C
      CALL DTABLE(1)
C
10    CONTINUE
      GOTO(5,15), JUMP
15    CONTINUE
      CALL LIST('(WO3.FOR)')
      END
```

FIGURE 10.5: WASTE OIL RECLAMATION PROBLEM, MODIFICATION #3

```
                         **********************************
                         *  SUMMARY FOR VECTOR PVALUE     *
                         **********************************


                              STANDARD   COEF. OF                                   COEF. OF     COEF. OF
PERIOD  COUNT        MEAN    DEVIATION  VARIATION      MINIMUM        MAXIMUM     SKEWNESS     KURTOSIS

   1     5000  2815.71900   1818.11800      64.57   -1834.04600    11838.67000        .55          .57
   2     5000  1841.81700   1942.75500     105.48   -3190.54000    11204.57000        .50          .33
   3     5000   867.91000   2127.63600     245.14   -4988.61500    10570.46000        .42          .04
                                                                               ( 12.24)     (    .62)
```

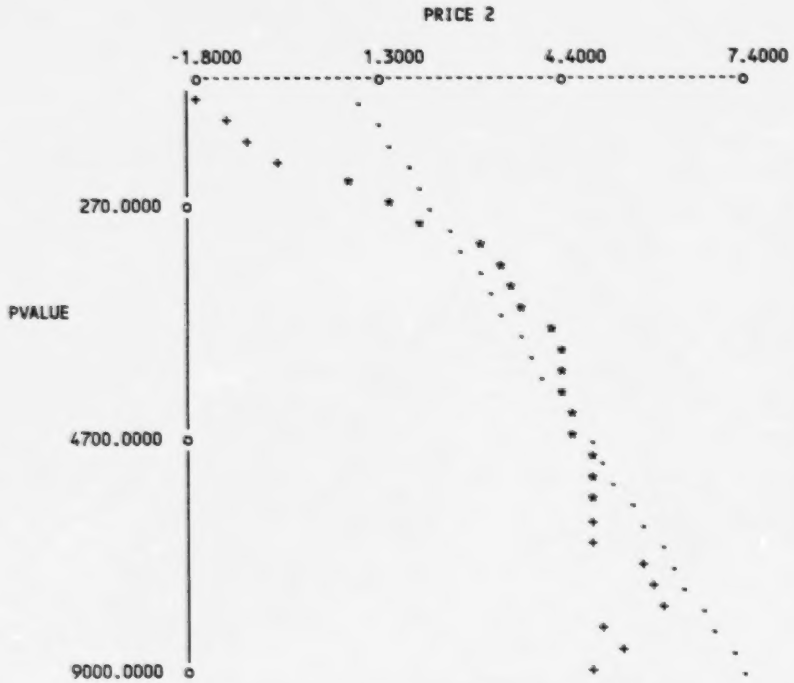
```
                         AVERAGE OF PVALUE

             870.0000       1500.0000       2100.0000       2800.0000
               o----------------o----------------o----------------o
           1  o                                                  *
              |
              |
              |
PERIOD     2  o                        *
              |
              |
              |
           3  o*
              |
              |
              |
```

FIGURE 10.6:

VECTOR OUTPUT OF THE WASTE OIL RECLAMATION PROBLEM, MODIFICATION #3

```
****************************************************
*    PLOT OF QUANTITY PRICE 2  VERSUS PVALUE       *
****************************************************
```



```
Point code: * = within 20% of maximum weight
            + = less than 20% of maximum weight
Dotted line is best fit, i.e., LINEAR MODEL
```

```
              SUMMARY OF FITTED EQUATIONS, WEIGHTED LEAST SQUARES

                                     A               B        R-SQUARE
LINEAR      : Y=A+B*X           .19180E+01      .60267E-03      .76
QUADRATIC   : Y=A+B*X*X         .28672E+01      .67051E-07      .44
CUBIC       : Y=A+B*X*X*X       .32092E+01      .77559E-11      .29
RECIPROCAL  : Y=A+B/X           .35563E+01      .14997E+03      .04
SQUARE ROOT: Y=A+B*sqrt(X)      .00000E+00      .00000E+00      .00
LOGARITHMIC: Y=A+ln(X)          .00000E+00      .00000E+00      .00
POWER       : Y=A*X**B          .00000E+00      .00000E+00      .00
EXPONENTIAL: Y=A*exp(BX)        .00000E+00      .00000E+00      .00
```

FIGURE 10.7:

PLOT OUTPUT OF THE WASTE OIL RECLAMATION PROBLEM, MODIFICATION #3

more, the vector version of the program runs much faster. However, if the present value distributions had been non-normal, then it would have been better to have used the approach of Figure 7.1.

Another change has been incorporated into the Waste Oil Reclamation Problem, Modification #3, i.e., the Plot Definition line which involves a vector, PVALUE(2). This leads to consideration of the conditions under which standard tables and plots can be used in connection with vectors. Standard tables and plots <u>cannot</u> be used within vector loops. Used outside of the vector loop, the vector can be entered in the standard table or plot call as a <u>specific</u> coordinate, e.g., PVALUE(2). In the Waste Oil Reclamation Problem, Modification #3, the Plot Definition line has been placed <u>after</u> Vector Loop Control Line #3. The plot produced from this program is shown in Figure 10.7.

EXERCISES

Which of the following are valid constructions in MOUSE?

1. `X1=PWSUM(PROFIT,.10,XLIFE,XININV)`

2. `DIMENSION ANBEN(30)`

3.
```
      DO 100 LOOP=1,TIME
         .
         .
         .
100   CONTINUE
      LOOP=1
```

4. `CALL TABLE(PROFIT,'(PROFIT)',LIFE,T10)`

5. `PRECISION T1(7,20),T2(7,10)`

6. `CALL VTABLE(SPEED,'(SPEED)',LAPS,T2)`

7. Assume, in the Modification #2 to the Waste Oil Reclamation problem, that there is a salvage value at the end of the project life which varies according to a continuous uniform distribution over the interval $4000 to $6000. Using an IF statement, what lines should be altered and/or added to the program?

The following two problems are <u>optional</u>. They show how MOUSE may be used as a more general purpose simulation language.

8. (This exercise involves the use of the IF and GOTO statements described in Chapter 4, Sections 4.6 and 4.7.) A rat is placed into a cage containing three doors. The first door leads to freedom, the second leads to a long tunnel, and the third leads to a short tunnel. Both tunnels, however, return the rat to the cage. The doors all look alike to the rat. The time spent traversing the first tunnel is exponentially distributed with a mean of 3 time units and that the time spent traversing the second tunnel is lognormally distributed with a mean of 6 and standard deviation of 2. Suppose that the rat's remaining time to live is normally distributed with a mean of 10 and standard deviation of 2. Write a MOUSE program that will, on the basis of the experience of 1000 rats, determine: (a) The probability that the rat reaches freedom before it dies, and (b) The probability distribution of the time the rat spends in trying to escape, regardless of whether it lives or dies.

9. (This exercise involves the use of the Vector loop and the Vector Table.) There are three square peg boards, each containing 100 holes arranged in a symmetric 10 x 10 pattern. Board #1 has a peg in each hole; the other boards are empty. Every hour, the following sequence occurs;

(1) 15 locations are chosen at random on the first board. If a peg is in one of the holes chosen, it is moved to its analogous position on Board #2.

(2) After the selections are made for Board #1, 5 locations are chosen at random on the second Board. If a peg is in one of the holes chosen, it is moved to its analogous position on Board #3.

Write a MOUSE program that plots, for 20 hours of operation, the number of pegs remaining on each board at the end of every hour.Use 100 20-hour simulations in your program..

CHAPTER 11

TROUBLESHOOTING, ERROR MESSAGES, AND HELP

11.1 Introduction

Regardless of the care taken in writing an uncertainty analysis program, errors are unavoidable. The five most common types are: (a) FORTRAN syntactical errors (i.e., errors caused by improper coding of the MOUSE program that are caught by the FORTRAN compiler), (b) MOUSE syntactical errors (i.e., errors caused by improper coding of the MOUSE program that are not caught by the FORTRAN compiler), (c) errors in confusing decimal numbers with integers, (d) errors in the input data, and (e) logical errors in the model. An example of a FORTRAN syntactical error is:

```
Y=(A+B)/(C+D
```

The error is in the fact that the parentheses are unbalanced, i.e., there are two left-hand parentheses in the equation but only one right-hand parenthesis, something that FORTRAN does not permit. The FORTRAN compiler (see Appendix A for a discussion of compilers and what they do) will catch this kind of error and, indeed, some FORTRAN systems (e.g., Microsoft see Appendix A, Section A.3) have an option whereby a source program can be checked for errors before it is compiled, thus saving considerable time since compiling is a time-consuming process.

All computers are limited as to the size of the numbers they can manipulate, the limitation varying with the particular computer. The multiplication of large numbers or the division of a number by another close to zero can easily exceed the capacity of a computer. Either output will be deleted (usually by printing asterisks where numbers are expected) or error messages (such as "FLOATING POINT ARITHMETIC OVERFLOW") will be sent to the user. This is most likely to occur when Vector Tables are employed since the arithmetic involves taking numbers to their fourth power; other MOUSE utilities that are susceptible to such overflow conditions include the Plot routine and the standard MOUSE Table. Fortunately, scaling (see Chapter 6, Section 6.6 and Chapter 8, Section 8.2) by the user will correct these errors.

11.2 CHECKER, A Utility for Catching MOUSE Syntactical Errors

An example of a MOUSE syntactical error is:

```
CALL TABLE(VALUE,'(VALUE)',DUM0,DUM1,TAB)
CALL TABLE(COST,'(COST)',DUM2,DUM3,TAB)
```

The error here is in the fact that the storage variable, TAB, has been used for two different tables. A FORTRAN compiler would not catch this error; the program might run successfully but bizarre results would be obtained (see Section 11.3).

A utility called "CHECKER" can be used to parse MOUSE program files and check for a number of MOUSE syntactical errors. To use CHECKER, simply type CHECKER and the following screen will appear:

```
*********************************************************
*                        CHECKER                        *
*        (MOUSE Source File Checking Program)           *
*                     VERSION 1.00+                     *
*          US ENVIRONMENTAL PROTECTION AGENCY           *
*              Written by ALBERT J. KLEE                *
*********************************************************
               ********************
               * INTERACTIVE MODE * <--
               * MUTE MODE        *
               ********************
        *************************************
        * ENTER NAME OF FILE TO BE CHECKED: *
        *                                   *
        * >> WO2.FOR                        *
        *************************************
```

If the program is found, CHECKER will parse the file for certain errors (see below) and, if any are found, will report them in a file called CHECKER.OUT. Selection is made in the "Interactive/Mute" menu by using the up or down arrow keys and then pressing the Enter key. In Interactive Mode, each time an error is found the line in which the error occurs is printed on the screen, after which the program will continue if the Enter key is pressed. In Mute Mode, only the number of lines scanned and the number of errors found to date are reported on the screen (this information also appears in the Interactive Mode). Interactive mode is useful for small source files since one frequently can identify the error when the erroneous line appears on the screen. However, for large source files the Mute Mode is recommended as otherwise the constant interruption would make the checking process lengthy and the user must be present for the program to continue.

CHECKER first examines the Dimension Block and, if the Dimension Block does not pass CHECKER's tests, either of the following messages will be reported:

```
ERROR IN DIMENSION LINE x
Parentheses are unbalanced, i.e.,
  Number of Left Parentheses  = y
  Number of Right Parentheses = z
 PROGRAM ABORTED - FIX DIMENSION BLOCK!
```

```
ERROR IN DIMENSION LINE x
Wrong number of commas, i.e.,
  Number of Paired Parentheses = y
  Number of Separating Commas  = z
The number of Separating commas
should be one less than the number of
Paired Parentheses.
PROGRAM ABORTED - FIX DIMENSION BLOCK!
```

If the Dimension Block successfully passes CHECKER's tests, the program goes on to check the following MOUSE line types:

Decimal Trace Line (i.e., CALL DTRACE)
Dummy Table Output Output Line (i.e., CALL DTABLE)
Empirical Variable Read Line (i.e., CALL READ)
Integer Trace Line (i.e., CALL ITRACE)
Matrix Read Line (i.e., CALL MREAD)
Name Line (i.e., CALL NAME)
Plot Output Output Line (i.e., CALL PLOT)
Program Listing Line (i.e., CALL LIST)
Sensitivity Control Line (i.e., CALL SEN)
Table Output (MOUSE) Output Line (i.e., CALL TABLE)
Table Output (USER) Output Line (i.e., CALL TABLE)
Vector Table Output Output Line (i.e., CALL VTABLE)

Also checked are all MOUSE probability and interest function calls, the Pass Control line, and the Specification Block User Control lines. A check is also made that all checked lines are in the proper order (e.g., that all Specification Block Control lines are placed _before_ MOUSE Control Line #4). Finally, CHECKER will ascertain whether the eight standard MOUSE Control Lines are correct and in the proper order and, if used, whether the three standard Vector Loop Control Lines are correct and in the proper order.

CHECKER's primary function is to see that dimensioned variables are properly defined and used; its secondary function is to insure that arguments to probability and interest calls are correct, although a number of other checks are made as well. A list of such checks are as follows:

AGAUL Mode greater than Median.
BGAU1 dummy variable not a decimal name.
BGAU2 dummy variable not a decimal name.
BGAUL1 dummy variable not a decimal name.
BGAUL2 dummy variable not a decimal name.
Decimal point omitted in number x.
Dimension Block dimension does not agree with first
  Matrix Read line dimension x.
Dimension Block dimension does not agree with second
  Matrix Read line dimension x.

Dimension Block dimension does not agree with Empirical Variable Read line dimension x.
Dimension Block dimension of x is not 6 more than Table Output line dimension of y.
Dimension of 999 exceeded in dimensioned name y.
Duplication of dimensioned variable y.
Duplication of dummy variable y.
Erroneous Column Output Line.
Erroneous Loop Control Line.
Erroneous Output Destination Line.
Erroneous Pass Control Line.
First Dimension Block dimension should equal 7 instead of x for Vector Table storage variable y.
First Dimension Block dimension should equal 100 instead of x for Plot storage variable y.
GAU standard deviation not greater than zero.
Illegal characters, illegal decimal name, or integer name exceeds 6 characters in PWSUM argument.
Illegal characters or integer name, or decimal name exceeds 6 characters in probability argument.
Illegal characters in PWSUM name.
Illegal characters or integer name, or decimal name exceeds 6 characters in interest argument.
Illegal decimal point in number x.
Illegal decimal point in PWSUM period argument.
Illegal integer name.
Illegal integer PWSUM name.
Improper range for correlation coefficient argument.
Improper range for probability argument.
Iteration number out of range.
Interest argument not greater than zero.
Interest period argument less than zero.
Length of 6 exceeded in dimensioned name y.
Length of 6 exceeded in name.
Length of 6 exceeded in PWSUM name.
Length of argument x illegally or unreasonably long.
Minus sign omitted in dimension x.
Missing or incorrect MOUSE Control Line x.
Missing or incorrect Vector Loop Control Line x.
Missing parenthesis in literal name y.
Missing quote in literal name y.
No decimal point in interest argument.
No decimal point in probability argument.
No dimension has been declared for storage variable x.
Non-integer for IPASS upper limit.
Non-integer or non-integer variable for LOOP range.
Number of trials greater than population.
Permissible length exceeded in literal name y.
Probability argument not a positive integer.
Probability argument not greater than zero.
Probability argument should be rounded off.
Probability arguments not in increasing order.

PWSUM interest argument not greater than zero.
PWSUM investment argument less than zero.
Second Dimension Block dimension should equal x
  instead of y for Plot storage variable z.
Seed Reset number not an odd integer.
Sensitivity Level number out of range.
Snap Interval number out of range.
Specification number not an integer.
The following two lines are in the wrong order:
This variable has been dimensioned more than once.
User and MOUSE dimensions confused.
Vector Loop Control Lines in wrong order.
x interest parameters found where y were expected.
x parameters found where y were expected.
x probability parameters found where y was expected.

CHECKER will identify suspect line orders, arguments, dimensions, numbers, names, and variables and, in most cases, the line in which the error occurred will also be printed.

11.3 A Case History

Figure 11.1 shows a short MOUSE program that was written by a colleague (an economist) of the author. The program compiled, linked, and ran (See Appendix A, Section A.1), without any difficulty. Output was obtained and is shown in Figure 11.3. A second table (that for variable SALES TX) was also produced but it was identical to that shown for the variable TAXREV in Figure 11.3. The tables were a complete disaster. The frequency distributions reported clearly were nonsense; other table entries made no sense either. For example, in Figure 11.2 the "Cumulative % Entries" column reached 190.10%. The "Number of Entries" column totaled 1901 instead of 1000. Also, a minor point but the name of the first table was reported as "TAXREV(T" instead of "TAXREV".

Judging by the number of entries that appeared in the table, it was clear that <u>two</u> tables were being combined. A quick glance at the program revealed the answer. My colleague had dimensioned only one table (T1) and had used the same dummy variables, AT1, BT1 in both tables. When CHECKER was run on the file, the following was reported:

```
                NAME OF FILE CHECKED: TAX1.FOR

 1.  Missing parenthesis in literal name '(TAXREV'
     in line CALL TABLE(TOTXRV,'(TAXREV',AT1,BT1,40,T1)

 2.  Duplication of dimensioned variable T1
     in line CALL TABLE(SALETX,'(SALES TX)',AT1,BT1,40,T1)
     and     CALL TABLE(TOTXRV,'(TAXREV',AT1,BT1,40,T1)
      PRECISION T1(46)
      COMMON/Q1/C80/Q2/JUMP/Q3/LOOP/Q4/SENS/Q5/IDATA/Q6/IOUT/Q7/IPASS/
     &/IRAN/Q9/IREAD/Q10/ISCR1/Q11/ISCR2/Q12/ISNAP/Q13/ITER/Q14/ITOG/Q
     &/IWRIT/Q16/IXXX/Q17/OK1/Q18/OK2/Q19/OK3/Q20/OK4/Q21/OK5/Q22/OK6
      CALL NAME('(TAX PROBLEM)')
      ITER=1000
5     DO 10 IXXX=1,ITER
      CONSUM=CUNI(95.,105.)
      PI=CUNI(120.,130.)
      PROFIT=CUNI(12.,14.)
      SALETX=.05*CONSUM
      XYTX=.10*PI
      PROTX=.10*PROFIT
      TOTXRV=SALETX+XYTX+PROTX
      CALL TABLE(TOTXRV,'(TAXREV',AT1,BT1,40,T1)
      CALL TABLE(SALETX,'(SALES TX)',AT1,BT1,40,T1)
10    CONTINUE
      GOTO(5,15), JUMP
15    CONTINUE
      END
```

FIGURE 11.1: ORIGINAL CASE HISTORY PROGRAM

```
      PRECISION T1(46),T2(46)
      COMMON/Q1/C80/Q2/JUMP/Q3/LOOP/Q4/SENS/Q5/IDATA/Q6/IOUT/Q7/IPASS/
     &/IRAN/Q9/IREAD/Q10/ISCR1/Q11/ISCR2/Q12/ISNAP/Q13/ITER/Q14/ITOG/Q
     &/IWRIT/Q16/IXXX/Q17/OK1/Q18/OK2/Q19/OK3/Q20/OK4/Q21/OK5/Q22/OK6
      CALL NAME('(TAX PROBLEM)')
      ITER=1000
5     DO 10 IXXX=1,ITER
      CONSUM=CUNI(95.,105.)
      PROFIT=CUNI(12.,14.)
      PI=CUNI(120.,130.)
      SALETX=.05*CONSUM
      XYTX=.10*PI
      PROTX=.10*PROFIT
      TOTXRV=SALETX+XYTX+PROTX
      CALL TABLE(TOTXRV,'(TAXREV)',AT1,BT1,40,T1)
      CALL TABLE(SALETX,'(SALES TX)',AT2,BT2,40,T2)
10    CONTINUE
      GOTO(5,15), LEAVE
15    CONTINUE
      END
```

FIGURE 11.2: CORRECTED CASE HISTORY PROGRAM

3. Duplication of dummy variable AT1
```
in line CALL TABLE(SALETX,'(SALES TX)',AT1,BT1,40,T1)
and     CALL TABLE(TOTXRV,'(TAXREV',AT1,BT1,40,T1)
```

4. Duplication of dummy variable BT1
```
in line CALL TABLE(SALETX,'(SALES TX)',AT1,BT1,40,T1)
and     CALL TABLE(TOTXRV,'(TAXREV',AT1,BT1,40,T1)
```

In short, CHECKER caught all of the MOUSE syntactical error in this program.

Figure 11.2 shows the corrected version of the program, and Figure 11.4 shows the results for the first table. The "improvement" is noticeable! This example is especially instructive since it shows that a program can be accepted and run by a computer without reservation and still produce incorrect answers.

11.4 MOUSE Run-time Error Checks

Because MOUSE is a modular system, i.e., a system consisting of a collection of functions and subroutines, it is necessary to transfer information between a main program (which the user writes) and these functions and subroutines. In most, if not all, high-level languages (such as FORTRAN, in which MOUSE is written) integer data are entered, stored, and written differently than are decimal data. If a function or subroutine is structured to receive or transmit decimal data it cannot substitute integer data or vice versa. Thus it is vital to adhere to the rules regarding decimal points in MOUSE (see Table 3.1 in Chapter 3). Failure to do so is a common cause of errors in MOUSE programs. Although a MOUSE program will successfully compile and run with such errors, incorrect results will be obtained.

Because of the vastly different manner in which integer and decimal data are stored in a computer, positive decimal variables without decimals are generally interpreted as extremely small positive numbers or else as negative numbers when passed in functions or subroutines. Integer variables, on the other hand, when mistakenly used with decimal points, are generally interpreted as numbers larger than 1000. MOUSE uses this fact to test decimal and integer numbers passed in certain functions or subroutines. For decimal numbers, MOUSE checks to see that the number is not less than a small number, OK6 (the distribution version of MOUSE sets OK6 equal to 0.0000000001 but this can be altered easily; see Appendix A, Section B.1). The call for the Poisson random variable is, for example, POI(mean). If the user mistakenly omitted the decimal point in the call, e.g., POI(5), MOUSE would detect this error easily since the mean of a Poisson distribution must be greater than zero. In this case, "zero" is considered by MOUSE to be the value of OK6. For the triangular

```
                    *************************************************
                    *    DISTRIBUTION FOR QUANTITY TAXREV(T         *
                    *************************************************


NUMBER OF ITERATIONS =  1000        MEAN =        21.95434          STANDARD DEVIATION =        11.75831
                                 MINIMUM =         4.75014   COEFFICIENT OF VARIATION, % =      53.55799
                                 MAXIMUM =        19.52732


     LOWER  NUMBER OF   PERCENT  CUMULATIVE  CUMULATIVE                DISTRIBUTIONS
     LIMIT   ENTRIES    ENTRIES  % ENTRIES   COMPLEMENT    * = FREQUENCY DISTRIBUTION     O = CUMULATIVE DISTRIBUTION

    4.3500     211.      21.10     21.10       78.90    **************O***
    4.8600     788.      78.80     99.90         .10    **************************************************************************
    5.3700       0.        .00     99.90         .10    *
    5.8800       0.        .00     99.90         .10    *
    6.3900       0.        .00     99.90         .10    *
    6.9000       0.        .00     99.90         .10    *
    7.4100       0.        .00     99.90         .10    *
    7.9200       0.        .00     99.90         .10    *
    8.4300       0.        .00     99.90         .10    *
    8.9400       0.        .00     99.90         .10    *
    9.4500       0.        .00     99.90         .10    *
    9.9600       0.        .00     99.90         .10    *
   10.4700       0.        .00     99.90         .10    *
   10.9800       0.        .00     99.90         .10    *
   11.4900       0.        .00     99.90         .10    *
   12.0000       0.        .00     99.90         .10    *
   12.5100       0.        .00     99.90         .10    *
   13.0200       0.        .00     99.90         .10    *
   13.5300       0.        .00     99.90         .10    *
   14.0400       0.        .00     99.90         .10    *
   14.5500       0.        .00     99.90         .10    *
   15.0600       0.        .00     99.90         .10    *
   15.5700       0.        .00     99.90         .10    *
   16.0800       0.        .00     99.90         .10    *
   16.5900       0.        .00     99.90         .10    *
   17.1000       0.        .00     99.90         .10    *
   17.6100       7.        .70    100.60        -.60    **
   18.1200     287.      28.70    129.30      -29.30    ***********************
   18.6300     465.      46.50    175.80      -75.80    *************************************
   19.1400     143.      14.30    190.10      -90.10    ************
   19.6500       0.        .00    190.10      -90.10    *
   20.1600       0.        .00    190.10      -90.10    *
   20.6700       0.        .00    190.10      -90.10    *
  OVERFLOW       0.        .00    190.10      -90.10    *
```

FIGURE 11.3: TABLE OUTPUT OF ORIGINAL CASE HISTORY PROGRAM

```
                    ******************************************
                    *    DISTRIBUTION FOR QUANTITY TAXREV    *
                    ******************************************


NUMBER OF ITERATIONS =  1000        MEAN =       18.80238             STANDARD DEVIATION =         .32029
                                 MINIMUM =       17.99231     COEFFICIENT OF VARIATION, % =       1.70347
                                 MAXIMUM =       19.47256

      LOWER  NUMBER OF  PERCENT  CUMULATIVE  CUMULATIVE                DISTRIBUTIONS
      LIMIT    ENTRIES  ENTRIES   % ENTRIES  COMPLEMENT   * = FREQUENCY DISTRIBUTION     O = CUMULATIVE DISTRIBUTION

  UNDERFLOW         1.      .10         .10       99.90  O*
    18.0000         1.      .10         .20       99.80  O*
    18.0520         6.      .60         .80       99.20  O******
    18.1040         5.      .50        1.30       98.70  *O****
    18.1560        15.     1.50        2.80       97.20  **O************
    18.2080        15.     1.50        4.30       95.70  ***O***********
    18.2600        24.     2.40        6.70       93.30  ****O*******************
    18.3120        28.     2.80        9.50       90.50  ******O********************
    18.3640        39.     3.90       13.40       86.60  ********O*****************************
    18.4160        37.     3.70       17.10       82.90  **********O*************************
    18.4680        45.     4.50       21.60       78.40  *************O*****************************
    18.5200        51.     5.10       26.70       73.30  ****************O********************************
    18.5720        50.     5.00       31.70       68.30  *******************O****************************
    18.6240        47.     4.70       36.40       63.60  **********************O**********************
    18.6760        51.     5.10       41.50       58.50  *************************O***********************
    18.7280        65.     6.50       48.00       52.00  *****************************O*******************************
    18.7800        51.     5.10       53.10       46.90  ********************************O****************
    18.8320        55.     5.50       58.60       41.40  ************************************O****************
    18.8840        46.     4.60       63.20       36.80  ***************************************O****
    18.9360        54.     5.40       68.60       31.40  ******************************************O*********
    18.9880        56.     5.60       74.20       25.80  *********************************************O********
    19.0400        47.     4.70       78.90       21.10  *********************************************   O
    19.0920        42.     4.20       83.10       16.90  ****************************************           O
    19.1440        43.     4.30       87.40       12.60  *****************************************            O
    19.1960        40.     4.00       91.40        8.60  ***************************************                 O
    19.2480        26.     2.60       94.00        6.00  *************************                                O
    19.3000        23.     2.30       96.30        3.70  ***********************                                    O
    19.3520        19.     1.90       98.20        1.80  *******************                                         O
    19.4040        12.     1.20       99.40         .60  ************
    19.4560         6.      .60      100.00         .00  *******
```

FIGURE 11.4: TABLE OUTPUT OF CORRECTED CASE HISTORY PROGRAM

distribution the call is TRI(a,b,c) where a and b are the lower and upper bounds, respectively, of the distribution. If the user entered TRI(3,5,9) instead of the correct TRI(3.,5.,9.), MOUSE would catch this since it would test to see that the quantity (c - a) was greater than OK6. The integer argument check to see if the integer passed exceeds 1000 is performed only when it is illogical that the number be this great. It is acceptable, for example, that the number of iterations exceeds 1000 but it is not acceptable that a table be dimensioned this large.

MOUSE provides 20 run-time error checks. If any of these errors are found by MOUSE, an error message is reported and the run is terminated. A list of these error messages can be found in Appendix C. MOUSE makes almost all of the integer versus decimal point checks necessary and, in addition, also tests for other common errors such as placing certain function parameters in the wrong order (e.g., in the continuous uniform distribution, the first parameter is the lower bound and the second is the upper bound and not vice versa), exceeding the permissible range of the correlation coefficient in the bivariate normal or bivariate lognormal distribution, decreasing the variance by more than 100% in the Sensitivity Control Line, etc. Other MOUSE requirements are also checked, such as the need to dimension tables 100 or more when MOUSE-dimensioned tables are used or having the number of iterations exceed the snap interval number.

The above error checks are static checks, i.e., they are made at the very beginning of the run (i.e., at IXXX=1) and, if any errors are found, the run is terminated. If the MOUSE call at IXXX=1 is correct, however, static errors will not be discovered. The interest rate in interest functions, however, is frequently supplied as a random variable, e.g.,

```
RATE=GAU(0.10,0.4)
      .
      .
      .
X1=COST*SCA(RATE,XLIFE,0.)
```

It is very possible, under these conditions, for RATE to be positive for values of IXXX=1 and negative at some other values. Interest functions, therefore, have a special error trap to catch such dynamic errors. Each call to the function is monitored, regardless of the value of IXXX, and, if negative interest rates or negative time periods are found, an error message will be reported, along with the value of IXXX at which the error was found, and the run will be terminated.

Although CHECKER will find and report the integer/decimal errors described above (and it will find them <u>before</u> compilation, thus saving much labor), if the arguments are in the form of variables rather than numbers, e.g., POI(VAR), CHECKER skips the integer/decimal check (other than checking to see that an integer or decimal variable name has been used) because it has no way of knowing what value these variables will take during a MOUSE run. The MOUSE run-time error checks, therefore, act as a kind of insurance.

11.5 The MOUSE Trace

As has been mentioned, when errors are incurred that are a result of an invalid instruction to the computer (i.e., errors in syntax), messages are reported to the user by the system's FORTRAN compiler that specify where in the main program such errors occurred. This, of course, provides a help to the user in tracking down the source of the error. CHECKER and the MOUSE run-time error messages also are of help in this regard. However, there are errors that can go undetected by the FORTRAN system, by CHECKER or by MOUSE. We might, for example, enter the value of a constant as 1000 instead of 10; to the FORTRAN compiler or to MOUSE this is not an error and it will obediently use the incorrect value in its calculations without sensing that anything is wrong. These sorts of errors, which can involve logic as well as data or parameters (e.g., a minus sign instead of a plus sign in an uncertainty equation or perhaps an incorrect uncertainty equation), can be quite difficult to find in complex problems. In fact, the user might not be aware that such errors exist and unfortunately accept the MOUSE output as delivered.

With regard to empirical variables or matrix data which are read into MOUSE, the echo listing automatically produced in every MOUSE run provides a convenient error check. One common method used to uncover other sorts of errors is to do a single sample calculation by hand to see whether the numbers produced are reasonable. This kind of "debugging" often can become tedious. Help is available, however, in the form of <u>Trace</u> <u>Lines</u>. A Trace Line has either of the following two forms:

```
           CALL DTRACE(dvariable,'(lname)')

or         CALL ITRACE(ivariable,'(lname)')
```

where "'lname'" is as defined for Table Output Lines. The only difference between the DTRACE and ITRACE Trace lines is that the former are for decimal variables, i.e., "dvariable", and the latter, i.e., "ivariable", are used for integer variables (see Chapter 3, Section 3.2).

To illustrate their use, Figure 11.5 shows the placement of Trace Lines into the Modification #2 of the Waste Oil Reclamation problem. (For clarity, the Trace lines are shown indented.) Since LIFE is an integer variable, an integer Trace line was used for this variable. The results are shown in Figure 11.6. A trace is produced at the following number of iterations: 1, 20, 50, and 100. At iteration = 1, all variables are evaluated at the average value of the input variables. Thus, the values of the output variables at iteration = 1 are their non-stochastic values (i.e., the values under certainty using the average values of the input variables). At all other iterations, the values produced are those as a result of a true random sample at those iterations. Considering the input data and the model itself, these are all reasonable values for the variables traced.

As can be seen, a vector can also be used in a Trace line, e.g.,

```
        CALL DTRACE(ANBEN(LOOP),'(ANBEN)')
```

Notice that the index LOOP must be used when vectors are the object of Trace lines.

The following rules apply to the placement of Trace lines in a MOUSE program:

1. A Trace line for a variable can appear only after the first appearance of that variable in a program;

2. Trace lines for variables (including vectors) that appear within vector loops must be placed within the loop itself. The values printed out for these variables, however, will always be for LOOP=1.

3. For the optimal operation of the trace, it is advisable to have at least 100 iterations, otherwise output at iterations 100, 50, and 20 may not be produced.

Trace lines are very useful for checking the performance of various MOUSE functions which require decimal numbers as arguments. If, by mistake, the decimal point is omitted, the function frequently will not indicate that an error has occurred but simply will return a value of zero. The use of a Trace line will indicate whether or not a reasonable number is being returned by the MOUSE function.

Any number of Trace lines can be used in a program but the presence of a single Trace line will override the user's Iteration line specification (which, as previously mentioned, should be 100 or greater in order to obtain output at iterations 20, 50, and 100). Since no tables or plots are created during a trace, the running time for a program containing one or more Trace lines is extremely short. Removal of the Trace lines restores the program to its full user-specified status.

Placing Trace lines into a MOUSE source program can be tedious. However, a MOUSE utility called TRACER can be used to automatically place Trace lines into a program or even remove

```
C
C WASTE OIL RECLAMATION PROBLEM MODIFICATION #2
C
      DIMENSION D1(3,2),D2(4,2),D3(4,2)
      DIMENSION ANBEN(15)
      DIMENSION VTAB(7,15)
      DIMENSION TAB(46)
      COMMON/Q1/C80/Q2/JUMP/Q3/LOOP/Q4/SENS/Q5/IDATA/Q6/IOUT/Q7/IPASS/Q8
     &/IRAN/Q9/IREAD/Q10/ISCR1/Q11/ISCR2/Q12/ISNAP/Q13/ITER/Q14/ITOG/Q15
     &/IWRIT/Q16/IXXX/Q17/OK1/Q18/OK2/Q19/OK3/Q20/OK4/Q21/OK5/Q22/OK6
      CALL NAME('(WASTE OIL RECLAMATION PROBLEM - MODIFICATION #2)')
      CALL READ(D1,3)
      CALL READ(D2,4)
      CALL READ(D3,4)
      ITER=5000
5     DO 10 IXXX=1,ITER
      CONTAM=DUNI(1.0,4.0)
           CALL DTRACE(CONTAM,'(CONTAN)')
      RAND1=CUNI(-10.0,10.0)
           CALL DTRACE(RAND1,'(RAND1)')
      PRICE1=EDIS(D1,3)
           CALL DTRACE(PRICE1,'(PRICE1)')
      SLOPE=ECTS(D2,4)
           CALL DTRACE(SLOPE,'(SLOPE)')
      PROD2=STEP(D3,4)
           CALL DTRACE(PROD2,'(PROD2)')
      RAND2=GAU(0.00,0.10)
           CALL DTRACE(RAND2,'(RAND2)')
      PRICE3=EBETA(6.5,8.0,9.5)
           CALL DTRACE(PRICE3,'(PRICE3)')
      PROD3=EXPON(18.0)
           CALL DTRACE(PROD3,'(PROD3)')
      XLIFE=DTRI(5.0,10.0,15.0)
           CALL DTRACE(XLIFE,'(XLIFE)')
      XININV=TRA(1600.0,2000.0,2200.0,2400.0)
           CALL DTRACE(XININV,'(XININV)')
      THRU1=10.0-CONTAM
           CALL DTRACE(THRU1,'(THRU1)')
      YIELD1=10.0-2.0*CONTAM
           CALL DTRACE(YIELD1,'(YIELD1)')
      PROD1=THRU1*YIELD1+RAND1
           CALL DTRACE(PROD1,'(PROD1)')
      GREV1=PRICE1*PROD1
           CALL DTRACE(GREV1,'(GREV1)')
      PRICE2=10.0-SLOPE*PROD2+RAND2
           CALL DTRACE(PRICE2,'(PRICE2)')
      GREV2=PRICE2*PROD2
           CALL DTRACE(GREV2,'(GREV2)')
      GREV3=PRICE3*PROD3
           CALL DTRACE(GREV3,'(GREV3)')
      LIFE=XLIFE
           CALL ITRACE(LIFE,'(LIFE)')
      DO 100 LOOP=1,LIFE
      YEAR=LOOP
           CALL DTRACE(YEAR,'(YEAR)')
      ANBEN(LOOP)=SCA(.12,YEAR,0.)*(GREV1+GREV2+GREV3-70.)
           CALL DTRACE(ANBEN(LOOP),'(ANBEN)')
      CALL VTABLE(ANBEN,'(ANBEN)',LIFE,VTAB)
100   CONTINUE
      LOOP=1
      PVALUE=PWSUM(ANBEN,0.08,LIFE,XININV)
           CALL DTRACE(PVALUE,'(PVALUE)')
      CALL TABLE(PVALUE,'(PRES VAL)',DUM0,DUM1,40,TAB)
10    CONTINUE
      GOTO(5,15), JUMP
15    CONTINUE
      CALL LIST('(WO2.FOR)')
      END
```

FIGURE 11.5: PLACEMENT OF TRACE LINES IN MODIFICATION #2 OF THE WASTE OIL RECLAMATION PROBLEM

```
TRACE AT IXXX = 1 (Average values):          TRACE AT IXXX = 20:

    CONTAM    =          2.50000                 CONTAM    =          1.00000
    RAND1     =           .00000                 RAND1     =         -3.46499
    PRICE1    =          8.45000                 PRICE1    =          8.00000
    SLOPE     =           .12500                 SLOPE     =           .17997
    PROD2     =         52.50000                 PROD2     =         64.25715
    RAND2     =           .00000                 RAND2     =           .05501
    PRICE3    =          8.00000                 PRICE3    =          7.34273
    PROD3     =         18.00000                 PROD3     =         21.00083
    XLIFE     =         10.00000                 XLIFE     =         10.00000
    XININV    =       2100.00000                 XININV    =       2022.62300
    THRU1     =          7.50000                 THRU1     =          9.00000
    YIELD1    =          5.00000                 YIELD1    =          8.00000
    PROD1     =         37.50000                 PROD1     =         68.53501
    GREV1     =        316.87500                 GREV1     =        548.28010
    PRICE2    =          3.43750                 PRICE2    =         -1.50923
    GREV2     =        180.46880                 GREV2     =        -96.97889
    GREV3     =        144.00000                 GREV3     =        154.20340
    LIFE      =               10                 LIFE      =               10
    YEAR      =          1.00000                 YEAR      =          1.00000
    ANBEN     =        639.90500                 ANBEN     =        599.76510
    PVALUE    =       4916.69700                 PVALUE    =       4553.93100
    RATRET    =           .38900                 RATRET    =           .37900


TRACE AT IXXX = 50:                          TRACE AT IXXX = 100:

    CONTAM    =          2.00000                 CONTAM    =          1.00000
    RAND1     =          1.26925                 RAND1     =         -2.13976
    PRICE1    =          7.50000                 PRICE1    =          8.00000
    SLOPE     =           .07756                 SLOPE     =           .07618
    PROD2     =         64.68204                 PROD2     =         50.17829
    RAND2     =          -.23779                 RAND2     =          -.02676
    PRICE3    =          8.23146                 PRICE3    =          7.48467
    PROD3     =          4.66683                 PROD3     =         29.73042
    XLIFE     =         10.00000                 XLIFE     =          9.00000
    XININV    =       2020.78800                 XININV    =       2183.47500
    THRU1     =          8.00000                 THRU1     =          9.00000
    YIELD1    =          6.00000                 YIELD1    =          8.00000
    PROD1     =         49.26925                 PROD1     =         69.86024
    GREV1     =        369.51940                 GREV1     =        558.88190
    PRICE2    =          4.74562                 PRICE2    =          6.15046
    GREV2     =        306.95620                 GREV2     =        308.61930
    GREV3     =         38.41480                 GREV3     =        222.52240
    LIFE      =               10                 LIFE      =                9
    YEAR      =          1.00000                 YEAR      =          1.00000
    ANBEN     =        722.27720                 ANBEN     =       1142.42700
    PVALUE    =       5899.13700                 PVALUE    =       8876.06500
    RATRET    =           .45000                 RATRET    =           .62400
```

FIGURE 11.6: OUTPUT OF THE TRACED PROGRAM OF FIGURE 11.5

them. (The Trace lines in Figure 11.5 were actually placed there by TRACER.) Simply type TRACER and you will be queried for the name of the source file and the output file. If the output file exists, you will be asked whether you wish to write over the file or not. The next query asks whether you wish to add Trace lines or to remove them. TRACER will satisfactorily place Trace lines in a source file in most cases, but there are two instances where you will have to make small corrections manually:

(1) If you have an equation that uses more than one line (i.e., where a continuation character is used in column six see Chapter 3, Section 3.2), TRACER will place the Trace line immediately after the first line. The line should be moved manually (using your text editor) to the last line of the continuation series.

(2) If there is an evaluation of a variable after an IF statement, e.g.,

```
IF(RAN(IRAN) .LT. 0.5) PRICE=CUNI(100.,2000.)
```

TRACER will not produce a Trace line for this variable, and one will have to be inserted manually.

In either case, however, TRACER will tell you if any Continuation lines or IF statements requiring Trace lines were found, and how many of them there were.

11.6 TRAP, A Utility For Preparing MOUSE Source Programs

The overhead information (i.e., Control, Dimension, Read, Specification, and Output Lines) required by MOUSE is sometimes complex, and users usually must resort to various chapters in this book, the Summary Tables in Appendix B or the one-line MSHELP program (described in the next Section) in order to refresh their memories as to the form in which it must be supplied. For example, Waste Oil Reclamation Problem, Modification #2, requires the following overhead lines even before the stochastic sampling takes place and the model is defined:

```
      DIMENSION D1(3,2),D2(4,2),D3(4,2)
      DIMENSION ANBEN(15)
      DIMENSION VTAB(7,15)
      DIMENSION PTAB(46),RTAB(26)
      COMMON/Q1/C80/Q2/JUMP/Q3/LOOP/Q4/SENS/Q5/IDATA/Q6/IOUT/Q7/IPASS/Q8
     &/IRAN/Q9/IREAD/Q10/ISCR1/Q11/ISCR2/Q12/ISNAP/Q13/ITER/Q14/ITOG/Q15
     &/IWRIT/Q16/IXXX/Q17/OK1/Q18/OK2/Q19/OK3/Q20/OK4/Q21/OK5/Q22/OK6
      CALL NAME('(WASTE OIL RECLAMATION PROBLEM - MODIFICATION #2)')
      CALL READ(D1,3)
      CALL READ(D2,4)
      CALL READ(D3,4)
      ITER=5000
5     DO 10 IXXX=1,ITER
```

In addition, the user must ultimately also prepare the following lines.

```
      CONTAM=DUNI(1.0,4.0)
      RAND1=CUNI(-10.0,10.0)
      PRICE1=EDIS(D1,3)
      SLOPE=ECTS(D2,4)
      PROD2=STEP(D3,4)
      RAND2=GAU(0.00,0.10)
      PRICE3=EBETA(6.5,8.0,9.5)
      PROD3=EXPON(18.0)
      XLIFE=DTRI(5.0,10.0,15.0)
      XININV=TRA(1600.0,2000.0,2200.0,2400.0)
      CALL VTABLE(ANBEN,'(ANBEN)',LIFE,VTAB)
      CALL TABLE(RATRET,'(INT ROR)',DUM2,DUM3,20,RTAB)
      CALL TABLE(PVALUE,'(PRES VAL)',DUM0,DUM1,40,PTAB)
      CALL LIST('(WO3.FOR)')
      DO 100 LOOP=1,LIFE
100   CONTINUE
      LOOP=1
10    CONTINUE
      GOTO(5,15), JUMP
15    CONTINUE
      END
```

Typing all of this information into a MOUSE source file clearly can be tedious. TRAP is a miniature preprocessor (TRAnsformation Preprocessor) that simplifies this task considerably. It is a stand-alone interactive program that, like IMP, is entirely separate and distinct from MOUSE. The following is an example, using Waste Oil Reclamation Problem, Modification #2, of a TRAP session.

```
          ******************************************
          *                 TRAP                   *
          *     (TRAnsformation Preprocessor)      *
          *            VERSION 1.00+               *
          *   US ENVIRONMENTAL PROTECTION AGENCY   *
          *       Written by ALBERT J. KLEE        *
          ******************************************

               *********************************
               * PREPARE MOUSE DATA FILE       *
               * PREPARE MOUSE SOURCE PROGRAM * <--
               *********************************

ENTER OUTPUT FILE NAME (not to exceed 12 characters, including
extension) or press Return key for default of PROGRAM.FOR:

                         >> WO2.FOR
```

If the file WO2.for already exists, you will be asked whether you wish to overwrite this file or not. The other option, "PREPARE

MOUSE DATA FILE", will be discussed in section 11.7. TRAP then asks whether you wish to have comment lines placed into the output file:

```
                    ******************************
                    * FILE WO2.FOR          | YES * <--
                    * EXISTS... OVERWRITE?  | NO  *
                    ******************************
                    *******************************
                    * DO YOU WISH TO HAVE    | YES * <--
                    * THE OUTPUT COMMENTED?  | NO  *
                    *******************************
```

TRAP then starts its queries with the Specification Block. There are three options: The first permits the user to modify any of the MOUSE default values for this block, the second accepts the default values (e.g., number of iterations = 1000, sensitivity level = 3, etc.), and the third produces a basic MOUSE shell consisting of the eight standard MOUSE Control Lines, a Name Line, and a Dummy Table Output Line. The menu is as follows:

```
          *****************************************
          * 1. MODIFY DEFAULT SPECIFICATION BLOCK * <--
          * 2. ACCEPT DEFAULT SPECIFICATION BLOCK *
          * 3. PRODUCE BASIC MOUSE SHELL          *
          *****************************************
```

The following example uses the "MODIFY" option.

```
     **************************************************
     *    SPECIFICATION AND MISCELLANEOUS SECTION     *
     * ---------------------------------------------- *
     * Code is as follows: Y=YES or press Return key  *
     * to skip the query or to accept noted default.  *
     **************************************************

  ENTER NUMBER OF ITERATIONS (or Return to default to 1000):
                         >> 5000

       80 COLUMN OUTPUT? (Y or Return to skip):
                         >>

      ENTER SENSITIVITY LEVEL (1/2/3 or Return to skip):
                         >>

       ENTER SNAP INTERVAL (or Return to skip):
                         >>
    SEND MOUSE OUTPUT TO THE SCREEN? (or Return to skip):
                         >>
```

```
****************************************************
* SPECIFICATION & MISCELLANEOUS SECTION CONTINUED *
* ---------------------------------------------- *
*   Y=YES or press Return key to skip the query.  *
****************************************************

 ENTER NEW RANDOM NUMBER SEED (or Return to skip):
                    >>

 ENTER NUMBER OF PASSES (or press Return to skip):
                    >>

ENTER LISTING FILE NAME, not to exceed 12 characters,
including extension (or Return to skip):
                    >> WO2.FOR
```

Note that the program shows you how to enter the required data. If you make an error, TRAP will notify you of the fact (by beeping) and ask you to reenter the information. In the Waste Oil Reclamation Problem, Modification #2, only the listing file and the number of iterations was specified, so only this information needed to be entered.

With regard to stochastic variables, tables, plots, vectors and vector tables, matrices, and data accumulations, the following menu provides these options.

```
*************************************
* 1. PROCESS STOCHASTIC VARIABLES   * <--
* 2. PROCESS TABLES                 *
* 3. PROCESS PLOTS                  *
* 4. PROCESS VECTORS/VECTOR TABLES  *
* 5. PROCESS MATRICES               *
* 6. PROCESS DATA ACCUMULATIONS     *
* 7. FINISHED                       *
*************************************
```

Only the queries for one stochastic variable (PRICE1), one table (PVALUE), and the vector (ANBEN) with its associated vector table are shown in the following example.

TRAP presents a menu of 23 probability distributions for stochastic variable sampling purposes. The following example shows the procedure for the stochastic variable PRICE1, which is generated from an empirical discrete probability distribution evaluated at three points.

```
**************************************************************************
*                  STOCHASTIC VARIABLE SAMPLING MENU                     *
*  1. Beta, Standard Form     9. Empirical Beta        17. Lognormal,    *
*  2. Binomial               10. Empirical Cts             Alternative Form *
*  3. Bivariate Lognormal    11. Empirical Discrete   18. Normal         *
*  4. Bivariate Normal       12. Erlang               19. Pascal/Geometric *
*  5. Continuous Uniform     1*. Gamma                20. Poisson        *
*  6. Discrete Trapezoidal   14. Exponential          21. Step Rectr     *
*  7. Discrete Triangular    15. Hypergeometric       22. Trapezoidal    *
*  8. Discrete Uniform       16. Lognormal            23. Triangular     *
*                                                                        *
*  NUMBER OF PROBABILITY FUNCTION DESIRED? (or Return to exit): >> 11    *
**************************************************************************
```

```
     Empirical Discrete Distribution - Example: NAME=EDIS(VAR1,5)
     Enter VARIABLE NAME, and DISTRIBUTION DIMENSION
     separated by commas or spaces, e.g., NAME,5
     (or Return to select new probability distribution):
     >> PRICE1,3
```

The next query involves Sensitivity Control Lines:

```
        *****************************************************
        * 1. SKIP SENSITIVITY CONTROL LINE                  *
        * 2. SELECT MEAN SENSITIVITY ANALYSIS               *
        * 3. SELECT STANDARD DEVIATION SENSITIVITY ANALYSIS *
        * 4. SELECT BOTH                                    * <--
        *****************************************************
```

Although the Waste Oil Reclamation Problem, Modification #2, did not involve Sensitivity Control Lines, for illustrative purposes this example will ask for both mean and standard deviation Sensitivity Analysis Lines (50% increase in the mean and 20% increase in the standard deviation):

```
             ENTER PERCENTAGE CHANGE IN MEAN: >> 50
     ENTER PERCENTAGE CHANGE IN STANDARD DEVIATION: >> 20
```

Although MOUSE requires decimal points in all probability functions (except for the three empirical types, continuous discrete, and step), TRAP will place trailing decimal points into the stochastic variable sampling lines if you choose to omit them in the TRAP input.

```
*********************************************************************
*                          TABLE SECTION                            *
* ----------------------------------------------------------------- *
*                      Sample Entry: COST,40                        *
*  where COST is variable to be tabled with 40 histogram intervals. *
* ----------------------------------------------------------------- *
* Note: Press Return key to exit to next section.                   *
*********************************************************************
```

```
ENTER VARIABLE NAME AND NUMBER OF INTERVALS, SEPARATED  BY A COMMA:
                         >> PVALUE,40

        ****************************************************
        *             VECTOR & VECTOR TABLE SECTION        *
        * -------------------------------------------------*
        *              Sample Entry: NETBEN,50             *
        *     where NETBEN is vector dimensioned at 50.    *
        * -------------------------------------------------*
        * Note: Press Return key to exit to next section.  *
        ****************************************************

       ENTER VECTOR NAME AND DIMENSION, SEPARATED BY A COMMA:
                          >> ANBEN,15

                        ********************
                        * TABLE THIS | YES * <--
                        * VECTOR?    |  NO *
                        ********************
```

The output, which is fully commented, from this TRAP example is shown in Figure 11.7. Provided with the TRAP output, all that remains in order to obtain a completed MOUSE source file is for you to enter the stochastic sampling lines and the model itself. Note that TRAP provides standard names for empirical stochastic variable, table, vector table, matrix, and plot dimensions (these are, respectively, VARi, TABi, VTABi, MTABi, PTABi, where the i's are successive integers). Also, for simplicity, TRAP makes the literal names equal to the variable names in Sensitivity Control lines, and in Table, Plot, and Vector Table Output lines.

11.7 Using TRAP To Prepare MOUSE Data Files

Option 1 on the first TRAP menu is for preparing MOUSE empirical variable data files:

```
**********************************
* PREPARE MOUSE DATA FILE        * <--
* PREPARE MOUSE SOURCE PROGRAM   *
**********************************
```

By default, TRAP writes data information to the file MOUSE.DAT, but if the latter already exists you will asked for permission to overwrite it or to select a new target data file name.

```
*********************************************************
* 1. DISCRETE EMPIRICAL VARIABLE                        * <--
* 2. CONTINUOUS EMPIRICAL OR STEP RECTANGULAR VARIABLE  *
* 3. QUIT PROGRAM                                       *
*********************************************************
```

```
C
C Empirical Variable Dimension Lines
C
      DIMENSION VAR1(3),VAR2(4),VAR3(4)
C
C Table Dimension Lines
C
      DIMENSION TAB1(46)
C
C Vector Table Dimension Lines
C
      DIMENSION VTAB1(7,15)
C
C Vector Dimension Lines
C
      DIMENSION ANBEN(15)
C
C Mouse Control Lines 1,2, and 3
C
      COMMON/Q1/CBO/Q2/JUMP/Q3/LOOP/Q4/SENS/Q5/IDATA/Q6/IOUT/Q7/IPASS/Q8
     &/IRAN/Q9/IREAD/Q10/ISCR1/Q11/ISCR2/Q12/ISNAP/Q13/ITER/Q14/ITOG/Q15
     &/IWRIT/Q16/IXXX/Q17/OK1/Q18/OK2/Q19/OK3/Q20/OK4/Q21/OK5/Q22/OK6
C
C Name Line
C
      CALL NAME('WASTE OIL RECLAMATION PROBLEM - MODIFICATION #2)')
C
C Empirical Variable Read Lines
C
      CALL READ(VAR1,3)
      CALL READ(VAR2,4)
      CALL READ(VAR3,4)
C
C Iteration Number Line
C
      ITER=5000
C
C Mouse Control Line 4
C
5     DO 10 IXXX=1,ITER
C
C Stochastic Variable Sampling Lines
C (and Sensitivity Analysis Lines, if specified)
C
      CONTAM=DUNI(1.,4.)
      RAND1=CUNI(-10.,10.)
      PRICE1=EDIS(VAR1,3)
        CALL SEN(PRICE1,'(PRICE1)',50.,0.)
        CALL SEN(PRICE1,'(PRICE1)',0.,20.)
      SLOPE=ECTS(VAR2,4)
      PROD2=STEP(VAR3,4)
      RAND2=GAU(0.,.1)
      PRICE3=EBETA(6.5,8.,9.5)
      PROD3=EXPON(18.)
      XLIFE=DTRI(5.,10.,15.)
      XININV=TRA(1600.,2000.,2200.,2400.)
C
C Vector Loop Control Lines With Vector Table Output Lines
C
      DO 100 LOOP=1,LIFE
      CALL VTABLE(ANBEN,'(ANBEN)',LIFE,VTAB1)
100   CONTINUE
      LOOP=1
C
C Table Output Lines
C
      CALL TABLE(PVALUE,'(PVALUE)',DUM1,DUM2,40,TAB1)
C
C Mouse Control Lines 5, 6, 7, and 8, With Listing Line
C
10    CONTINUE
      GOTO(5,15), JUMP
15    CONTINUE
      CALL LIST('(WO2.FOR)')
      END
```

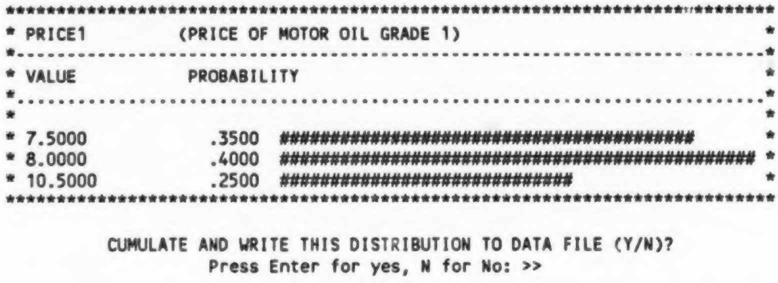
FIGURE 11.7: OUTPUT FROM SAMPLE TRAP RUN FOR WASTE OIL RECLAMATION PROBLEM, MODIFICATION #2

Until you elect the option to quit, you will be returned to this menu after every variable information set has been entered.

The processing of empirical variable data information proceeds in three steps:

1. Entering a Name Line;
2. Entering the values of the variable;
3. Entering the probability of each value (for discrete empirical variables) or the probability between each value (for continuous empirical and step rectangular variables).

If the total probability sums to less than 0.9999 or greater than 1.0001 (indicating that you may have made an error or else your probability rounding is not equal to this level of precision), you will be asked whether TRAP should adjust the total probability to one or abort processing the information about the variable. (If the total probability sums to less than 0.9999 or greater than 1.0001, you will not be queried, but the adjustment is automatically made if the sum is not one.) A summary graph of the adjusted probabilities is also displayed, e.g.,



```
******************************************************************
* PRICE1       (PRICE OF MOTOR OIL GRADE 1)                      *
*----------------------------------------------------------------*
* VALUE        PROBABILITY                                       *
*................................................................*
*                                                                *
* 7.5000         .3500 ###################################       *
* 8.0000         .4000 ########################################  *
* 10.5000        .2500 #########################                 *
******************************************************************

        CUMULATE AND WRITE THIS DISTRIBUTION TO DATA FILE (Y/N)?
                  Press Enter for yes, N for No: >>
```

11.8 MSHELP - Another MOUSE Help Utility

MOUSE overhead information must be supplied <u>exactly</u> as specified in this Manual or else errors will result when an attempt is made to run a program. In addition, MOUSE has many probability and interest calls available, and users usually must either resort to various chapters in this book or, as a better alternative, the Summary Tables in Appendix B in order to refresh their memories as to the form in which all of this information must be supplied. Neither of these alternatives may be convenient, however. A utility that provides another alternative is MSHELP. This is on-line, memory resident utility (about 30K of memory is required) that you can call up at any time while editing a MOUSE source program via a "hot key", i.e., Alt-H. This provides access to six different screens of information as follows:

(1) Mouse Control, Pass Control and Vector Loop Control Lines
(2) Mouse Interest Functions
(3) Mouse Probability Functions
(4) Mouse Probability Calls, (Truncated and Composite)
(5) Mouse Line Types (Dimension, Read, & Specification Blocks)
(6) Mouse Line Types (Program Block, Listing, Name, & Review Lines)

This is the same information contained in the summary tables of Appendix B. The information appears in a window that temporarily overwrites your file (if you are editing a file), and disappears when the escape key is pressed. You can page up or down among the screens, or transfer immediately to any selected screen.

MSHELP involves two files: MSHELP.COM and an auxiliary file, MOUSE.HLP. Both files should be placed into the same subdirectory on your hard disk, and your AUTOEXEC.BAT file must contain an appropriate PATH statement if you do not plan to edit your MOUSE source programs from this subdirectory. MSHELP is installed merely by typing "MSHELP". This is a memory resident program, but it can be removed simply by typing "MSHELP /U". The /U parameter uninstalls the program from memory. Note that MSHELP should be the last memory resident program installed in your computer.

Finally, if the default hotkey of Alt-H conflicts with any other hotkey, it may be changed by calling MSHELP as follows:

```
MSHELP /H#
```

where # is any alphanumeric key, the minus sign key or the equals sign key, e.g., MSHELP /HT. In this example, the hot key becomes Alt-T.

EXERCISES

Which of the following three sets of Trace lines, if any, are valid?

1.
```
CALL DTRACE(COST,'(COST)')
CALL ITRACE(NUMBER,'(NUMBER)')
```

2.
```
    DO 100 LOOP=1,LIFE
    YEAR=LIFE
    X1=GAU(100.,50.)
    X2(LOOP)=X1*SCA(.08,YEAR,0.)
100 CONTINUE
    LOOP=1
    CALL DTRACE(X2(LOOP),'(X2)')
```

3.
```
CALL DTRACE(PRICE,'(PRICE)')
CALL DTRACE(QUANT,'(QUANT)')
CALL DTRACE(X6,'(STOCK)')
```

CHAPTER 12

FITTING PROBABILITY DISTRIBUTIONS TO SAMPLE DATA

12.1 Introduction

An important part of uncertainty analysis is the determination of the appropriate probability distributions for the stochastic input variables. There are two basic approaches: (a) fit a probability distribution to sample data, and (b) elicit a subjective probability distribution. These approaches are not mutually exclusive; if the available data are few in number or of poor quality, then a combination of both methods may be used. This chapter is concerned with the first of these methods, i.e., fitting a probability distribution to sample data.

If the functional form of a probability distribution is known, the distribution is completely defined by its parameters. For example, given its mean and standard deviation, we need know nothing more about a normal distribution in order for MOUSE to generate random variables from the distribution. If, for example, the mean is known to be 5 and the standard deviation is known to be 2, then a call to GAU(5.,2.) in the MOUSE program is all that is required. If we postulate that the distribution is Poisson, then a call to POI(5.) is indicated. For some distributions, such as the beta, binomial, uniform, gamma, erlang, and pascal/geometric, the parameters of the distribution may be calculated directly from statistics such as the mean and standard deviation of the distribution. For example, the mean and standard deviation of a binomial distribution are given as:

$$\text{mean} = np \qquad [12.1]$$
$$\text{std} = [np(1 - p)]^{0.5} \qquad [12.2]$$

Solving Equations 12.1 and 12.2 for the binomial parameters p and n we obtain:

$$p = (\text{mean} - \text{std}^2)/\text{mean} \qquad [12.3]$$
$$n = \text{mean}^2/(\text{mean} - \text{std}^2) \qquad [12.4]$$

Substituting the values from the previous example of mean = 5 and std = 2 in Equations 12.3 and 12.4 we obtain p = 0.2 and n = 25. A call to BIN(0.2,25.) is all that is required then for MOUSE to generate random values from the distribution.

Some cases do pose more substantial difficulties. Consider the two-parameter Weibull distribution, for example, a distribution of considerable importance in life testing and reliability studies:

$$f(x) = (c/b)(x/b)^{c-1}\exp[-(x/b)^{c}] \quad [12.5]$$

where b is a scale parameter, and c is a shape parameter. The mean and standard deviation of the Weibull distribution are:

$$\text{mean} = bG[(1/c) + 1] \quad [12.6]$$
$$\text{std} = (b^{2}G[(2/c) + 1] - \text{mean}^{2})^{0.5} \quad [12.7]$$

where G is the gamma function. As can be seen, the gamma function is involved in the determination of the parameters of the Weibull distribution from its mean and standard deviation. To evaluate the gamma function in equations 12.6 and 12.7, however, we first must know the value of c (which we are trying to find). Accordingly, there is no direct algebraic solution for its parameters. Other methods (see Ruzicka, 1962), primarily of a graphical or trial-and-error nature, are generally used for this purpose. Some distributions, such as the hypergeometric, have more than two parameters so that knowledge only of the mean and standard deviation is not sufficient to estimate the parameters. For most distributions, however, the parameters may be estimated by simple statistical quantities readily calculated from sample data; once the parameters of a distribution are determined, they may be used with the standard MOUSE classical probability functions to generate the required random variates for the uncertainty analysis.

The problem now arises as to which form of probability distribution should be selected. Knowing certain sample statistics or facts about the sample data may help us to eliminate certain possibilities. Negative data, for example, precludes the exponential distribution (which only is defined for zero or positive values of the variable); for a lognormal distribution the median must be greater than the mode, etc. Graphing the sample data may provide insights as to the reasonableness of a postulated distribution but the truth of the matter is that, unless we have a great deal of sample data (200 or more points), it is difficult to determine the shape of a distribution with very much confidence simply by viewing a histogram of the sample data. This is where statistical procedures known as "goodness-of-fit" tests enter into the picture.

12.2 Goodness-Of-Fit Tests

A goodness-of-fit test sets up a null hypothesis about the identity of a probability distribution that has been fitted to a set of sample data. Since the alternative hypothesis is broad, rejection of the null hypothesis does not tell us what the distribution is, only what it is not. Furthermore, goodness-of-fit tests based upon estimates of parameters from sample data are only approximate. Like all statistical tests, they require that the user select a level of error called the "level of significance", i.e., the probability that the null hypotheses will be rejected when it is true. From a decision-theoretic point of view, this level of error should be selected according to the consequences of falsely rejecting the null hypothesis. Unfortunately, these consequences are hardly ever known and are difficult to ascertain. The classical statistical approach to goodness-of-fit tests, in which strict adherence is made to significance levels selected beforehand, therefore, makes little sense for the purposes of uncertainty analysis. They can be used, however, in conjunction with other information (objective or subjective) we may have about the distribution in question for we presumably have examined the data, not only by calculating statistics such as mean and standard deviation, but also by looking at histograms of the plotted data. In addition, we may also have subjective a priori opinions about the form of the distribution (see Section 12.3).

Another reason why caution should be used with goodness-of-fit tests is that some truncated distributions very closely approximate data sets that come from very different distributions. For example, data sets from exponential distributions can be fitted quite well to a normal distribution since the portion of a normal distribution one-half standard deviation to the left of its mean looks very much like an exponential distribution. By looking at a plot of the fitted versus the original data, however, it would be clear that the data were not symmetric and that therefore the full normal distribution would not be appropriate. (The truncated normal distribution - see Chapter 4, Section 4.7 - might be appropriate but the truncation point would have to be determined, perhaps simply by equating it to the minimum of the data set.) A goodness-of-fit test in an uncertainty analysis context, therefore, can be used to lend support, mitigate or alter the suspicions we already may have about the data set in question, but the final decision is up to the decision-maker himself.

The two best-known goodness-of-fit tests are the chi-square and Kolmogorov-Smirnov (K-S) tests. The former is specifically designed for use with discrete distributions, while the latter is intended for use with continuous distributions. (See Gibbons, 1976.) The chi-square test is based upon vertical deviations between the observed and hypothesized histograms of the data whereas the K-S test is based upon vertical deviations between the observed and hypothesized cumulative probabilities. There is another test, referred to here as the "Foutz test", which, although similar to the K-S test, is not as well-known. (See Foutz, 1980.) The Foutz test is also based upon vertical deviations between the observed and hypothesized cumulative probabilities but the Foutz test compares a much less restricted set of probabilities than does the K-S test. In simulations by the author, the Foutz test performed better than the K-S test for the classical distributions tested (i.e., continuous uniform, normal, lognormal, exponential, beta, gamma, and erlang). One disadvantage is that it is much more complex to calculate but in IMP, the computerized system for determining probability distributions from sample data, this is not a concern for the user.

12.3 Subjective Determination Of Probability Distributions

In the previous sections of this chapter, two basic approaches to the determination of the probability distributions of input stochastic variables were defined. The remainder of this chapter is concerned with the latter, i.e., the subjective determination of such distributions. The importance of the subjective approach increases as the quantity or quality of the historical or sample data available decreases; its value increases as the expertise of the estimator increases.

The basic assumption underlying the subjective approach is that the analyst knows more about the input variable in question than simply some measure of central tendency or "best" estimate. Any subject matter specialist has some store of knowledge, albeit "fuzzy" at times, that can be translated into some sort of probability distribution. The "fuzzier" such knowledge, of course, the greater the uncertainty reflected in the probability distribution so derived. Furthermore, the approach is not without risks since people are not always accurate assessors of probabilities; sufficient experimental evidence exists that estimation biases do exist.

The main emphasis now is to review the methods whereby we can obtain subjective probability distributions that most closely reflect the assessor's beliefs about the distribution in question and, at the same time, to minimize estimation biases. (For a review of these methods and the issues involved, see Huber, 1974.)

12.4 The Variable Interval Method

The variable interval method involves the identification of the points on a continuum that correspond to the endpoints of intervals containing specified proportions of the probability distribution. ("Probability distribution", as used here, means the probability density function.) This is usually explained in the form of a dialogue between a questioner (Q) and the analyst (A):

Q: To help us get started, what, in your view is the lowest possible unit price for steam?

A: Well, it hardly seems likely that the price could ever go below $2.

Q: How about the highest price?

A: I can't see it ever going above $18.

Q: Let's talk about more likely prices. What is the unit price of steam such that the actual price is just as likely to fall above that number as below it? Or, put another way, what is the steam price that you would bet even money on for being either exceeded or not exceeded?

A: Well, I'm not really sure but I suspect that it would be about $10.

Q: All right. Let's consider now how likely it is that other unit steam prices can occur. For the moment, forget about prices above $10 and let us concentrate on prices below that figure. What is the steam price that would divide this lower region (the region below $10) into two equal parts such that it would be just as likely that the actual price would fall below this price as it would to fall between this price and $10?

A: Maybe $8.

In the course of the above dialogue, the questioner has begun subdividing the continuum of steam prices into intervals of variable width containing equal proportions of the probability distribution, i.e.,

```
          .25        .25          .50
2-------------8----10------------------18
```

The process would continue, subdividing the quartiles on the left and proceeding afterwards to the right side of the $10 price. It should be mentioned that the process also involves checks for consistency:

> Q: You have, in effect, specified that there is a one in four chance that the price will fall between \$2 and \$8, and that there is a one in four chance that the price will fall between \$8 and \$10. Do you really think that these two outcomes are equally likely?

After considering this question, the analyst will often make revisions to his estimates. As the analyst gains experience with this technique, the revisions will become smaller. This example is a version of a psychological scaling technique, known as the "method of bisection", in which the intervals of the continuum are bisected with regard to probability.

Although a probability distribution is ultimately obtained by the variable interval method, the responses of the analyst are not probabilities but units of a quantity or measure with which he is familiar. By altering the questions slightly, the technique can be used to elicit cumulative probability distributions which, of course, are the forms required for direct use by MOUSE. However, studies have shown that people prefer the probability function form to the cumulative probability function form.

12.5 The Fixed Interval Method

In this method the continuum is divided into equal intervals and the analyst is asked to estimate the probabilities (or odds) associated with each interval. Using the example of Section 12.4, let us assume that the questioner and the analyst have agreed to work with intervals as follows:

2----4----6----8----10----12----14----16----18

The dialogue might proceed as follows:

> Q: Which of these outcomes is least likely to occur?
>
> A: Well, I'd say that a steam price of more than \$16 is the least likely.
>
> Q: OK. Now how likely is that outcome? What probability would you assign to it?
>
> A: Hmmmm. Say about one chance in twenty or 5%.
>
> Q: Let's hold that thought for a moment. Which of the remaining events is now least likely, and what do you think its probability is?
>
> A: That's hard to say. I kind of feel that a price of \$2 to \$4 is just as likely as one of \$14 to \$16; maybe a 1 in ten chance for either.

What the questioner has determined so far is:

```
 .10                              .10   .05
2----4----6----8----10----12----14----16----18
```

The questioning would proceed as before, with review and feedback to iron out any inconsistencies that might arise (such as the probabilities adding up to more than 1).

The fixed interval method deliberately highlights events of low probability and so tends to mitigate somewhat the problem of obtaining subjective probability estimates with too small a variance, a characteristic of subjective methods.

12.6 Other Methods

For certain probability distributions, such as the binomial, some investigators (see Winkler, 1967) have used an "equivalent prior sample" method where the analyst assessing the parameters of a probability function is asked for an r (successes) and an n (sample size) that, as a pair, represents his estimate of the population parameters and his uncertainty about his estimate:

> Q: You have some knowledge of the proportion of defects produced by this process. Can you give me two numbers, r and n, such that your knowledge would be roughly equivalent to having observed r defects in a random sample of n items from the production line? Remember, the ratio of r/n should be close to your assessment of the proportion of defects and that, for a given ratio of r/n, a larger n corresponds to a greater confidence in your assessment.
>
> A: Well, I can imagine a sample of 20 showing 1 defect so make $r = 1$ and $n = 20$.

Since the parameters of a binomial distribution are n (= 20 here) and p (= r/n = .05 here), the distribution required has been determined. The method is suitable only for discrete distributions for which knowledge of a postulated sample provides a unique determination of its parameters. Unfortunately, such distributions are not common.

By modification of the equivalent prior sample technique, the parameters of a standard beta distribution can be determined.

The method depends upon the fact that, in Bayesian terms, the conjugate distribution to the beta is the binomial distribution. The standard beta distribution (i.e., defined over the interval 0 to 1) with parameters a = 1 and b = 1 happens to be the continuous uniform distribution over the interval 0 to 1. Such a distribution implies complete lack of information since all values of the distribution are equally likely. In the Bayesian approach to be described (see Bierman, et al, 1961), this is considered the "prior distribution". The mean of the standard beta distribution is $a/(a + b) = r/n$ where $n = a + b$. If we obtain a sample from such a distribution, the application of Bayes Theorem tells us that the revised sample parameters are:

$$r' = 1 + r \qquad [12.1]$$
$$n' = 2 + n \qquad [12.2]$$

where r' and n' refer to the revised parameters, and r and n refer to the equivalent prior sample parameters. Thus, for our above sample if it were a standard beta distribution that was being estimated,

$$r' = 1 + 1 = 2$$
$$n' = 2 + 20 + 22$$

Accordingly, the parameters of the elicited standard beta distribution would be $a = r' = 2$, and $b = n' - r' = 22 - 2 = 20$.

Another method involves a series of bets or gambles. For example, the expected value (E) of a gamble is:

$$E = pW + (1 - p)L \qquad [12.3]$$

where p is the probability of winning, W is the amount to win, and L is the amount to lose. To enter this gamble, the analyst pays a sum, B, determined by the analyst. The analyst, therefore, presumably considers the expected value of the bet to be worth the amount B to him. If W and L are known, then p can be determined. For example:

> Q. I'll bet you that the steam price is between \$4 and \$5. If I am right I will pay you \$25; if wrong you pay me \$10. To take this gamble, however, you have to pay me something. How much will you pay me to take this gamble?
>
> A. I'll take that bet for \$15.

From Equation 13.3 and letting E = \$15, p (which is, in effect, the probability that the analyst thinks that the steam price lies between \$4 and \$6) can be calculated, i.e.,

$$15 = p(25) + (1 - p)(10) \text{ or } p = 1/3$$

The probabilities for other intervals are determined in a similar manner.

The difficulty with the equivalent prior sample method is that it is useful only when the form of the probability distribution is known. This begs the question since the emphasis in this chapter is on the subjective determination of the probability distribution, which includes not only its parameters but its form as well. The betting or wagering method depends upon a number of assumptions, e.g., that the analyst has a linear utility for money and that decision makers tend to maximize expected values. Furthermore, the method requires responses to three stimuli (W, L, and the outcome that decides the bet, in this instance the steam price interval of $4 to $6 postulated). For these reasons, these alternative approaches are not recommended by the author.

12.7 Improving Elicitation Methods

An important consideration involved in probability elicitation methods is how such methods can be made more accurate. There are three basic ways in which greater accuracy may be achieved: (1) train the estimators, (2) improve the elicitation technique and (3) aggregate the judgments of several estimators. Training includes the improvement of the statistical knowledge of the analysts (especially with regard to the concepts of probability distributions), review with the analysts of the experimental evidence that suggests the presence of bias in the tails and understating of the variance of subjectively elicited distributions, and actual practice in such elicitation, including feedback for consistency. With regard to biases, there is substantial evidence that people tend to rely primarily on specific information, even when it is unreliable, and that they give insufficient weight to general information. The result of this bias is to shift the entire elicited probability distribution to the left or to the right. Consider, for example, the following problem. A man has been selected at random from the adult population and described as "meek, tidy, and with a passion for detail". The question is then asked, "What odds would you give that this man is a librarian rather than a farmer?" The immediate intuitive answer is that the man in question is very likely to be a librarian. However, the population from which the man has been randomly drawn is the whole adult population and, in this class, there are many more farmers than librarians. Adjusting the estimate to take this into consideration would yield a much lower probability of the man being a librarian than just the intuitive guess. Some other facts about biases and understatement of the variance of the subjective probability distribution:

a. Recent information, being more available than old information, if often given much too much weight.

b. Initial estimates may serve as "anchors" for subsequent responses. Analysts, for example, seem to produce a central bias when they start the determination of a subjective probability distribution by focusing on the median or the mean of the distribution. In such cases, it is better to start first by looking at events with smaller probabilities (e.g., the tails of the distribution).

c. Laymen and experts alike exhibit considerable overconfidence. In general, the observed surprise rate is about 50% instead of 20%, i.e., when people are "80% sure", they are wrong about 50% of the time.

The elicitation method can be improved by the motivation of the participants and the rigid structuring of the elicitation (often referred to as the "encoding") procedure. An excellent discussion of this second approach can be found in Stael von Holstein (1972). Such rigid structuring, however, does have its disadvantages. It takes time and it often makes the probability elicitation procedure seem more complex than it really is. With training and practice, however, there is no reason why the analyst should not be able to derive acceptable subjective probability distributions by himself, using tools no more complicated than pencil and graph paper.

The third approach is to aggregate the judgments of several estimators, either mathematically or by group decision procedures. Mathematical methods include combining the individual probability distributions by means of some sort of weighted average. There are at least three approaches to the problem of weights:

1. Equal Weights: In this case the decision-maker has no reason to think that there is much difference in ability among the analysts, so he is willing to assign equal weights.

2. Weights Based on Previous Experience: This method may be subjective or objective, depending upon the quantity and quality of hard data obtained from prior experience with the analysts.

3. Weights Proportional to a Self-Rating: Have each analyst rate himself on a scale from 0 to M (conveniently M is either 10 or 100). Then assign a weight to each analyst proportional to his self-rating, where the constant of proportionality is such that the weights sum to one (the constant is obviously the reciprocal of the sum of the

self-ratings). The rationale behind this rule is that although a person may be able to make an assessment of a probability distribution, the degree of belief in that assessment may vary from analyst to analyst. Also, although an analyst may be an expert in a field, his expertise may vary from topic to topic within it.

Any of these weighted approaches are easy to implement in MOUSE. Consider the CONTAM variable in the Waste Oil Reclamation Problem. Assume that three analysts have been polled for their probability distributions for this variable and further that their respective distributions, A1, A2, and A3 are of the three-point empirical discrete form. Suppose that we have weighted the experts in the ratio 0.2, 0.2, and 0.4, respectively. The CONTAM call in MOUSE would appear as:

```
CONTAM=0.2*EDIS(A1,3)+0.2*EDIS(A2,3)+0.4*EDIS(A3,3)
```

Group Decision approaches can be of several types. In the first, termed "Group Reassessment" (GR), the analysts meet as a group and discuss the matter, ultimately arriving at a group consensus. This is, of course, nothing more than the familiar "Committee", "Panel" or "Workshop" method. The outcome of the GR method is apt to be a compromise among divergent views, arrived at under the undue influence of those with the greatest supposed authority or even merely the loudest voice. The group decision is affected on the one hand by the bandwagon effect of majority opinion and on the other by the unwillingness to abandon publicly expressed opinions. One way to reduce these effects is to allow each analyst to reassess his judgment in the absence of the others, a procedure known as Group Feedback and Individual Reassessment (GFIR). An interesting version of GFIR is the Delphi technique (Dalkey, 1975). In general, Delphi procedures have three features:

1. Anonymity: During a Delphi sequence the analysts are not made known to each other;

2. Controlled Feedback: The exercise is conducted in a sequence of rounds between which a summary of the results of the previous round is communicated to the participants;

3. Statistical Group Response: The procedure includes a response prepared by the administrator that includes the opinions of each member of the group, summarized by measures of central tendency (means, medians, etc.) and variability (quantiles, variances, ranges, etc.)

The most common form of Delphi is the paper and pencil version in which an administrator designs a questionnaire that is sent to each of the analysts. After the questionnaire is returned, the administrator summarizes the results, which becomes the basis for a new questionnaire for the group. This constitutes a "round". The analysts are given at least one opportunity to reevaluate their original subjective probability distributions based upon examination of the responses of the others. In general, those analysts whose responses are far from the mean of the group are asked to state their reasons why their responses are significantly different from the majority. In actual experience, placing an onus of justifying relatively extreme positions has the effect of causing those without strong convictions to move their distributions closer to the majority opinion, while those who feel that they have a good argument retain their original estimate and defend it.

TABLE 12.1

SUMMARY OF CONSENSUS TECHNIQUES

	Mathematical Combination	Group Reassessment	Group Feedback And Individual Reassessment
Can be implemented via mail questionnaire	yes	no	yes
Pressure to arrive at consensus	no	yes	yes
Presence of other adverse social pressures	no	yes	no
Consensus can be manipulated by decision maker	no	no	yes
Degree of difficulty for decision maker	low	low	high
Degree of difficulty for analysts	low	low to high	medium
Incorporates feedback	no	yes	yes
Analysts can be independently weighted	yes	no	yes
Introduction of new factors previously unthought of by some analysts	no	yes	partially
Commitment to group consensus	low	high	high

Table 12.1 compares the characteristics of the Mathematical Combination, Group Reassessment, and Group Feedback and Individual Reassessment procedures. As can be seen, no one technique dominates the others. The primary weakness of the Mathematical Combination technique is that it lacks feedback from the other analysts and is also low on commitment to group consensus, an important factor with regard to "selling" or otherwise implementing an uncertainty analysis. The primary advantage of Group Reassessment is the possibility of introducing new factors or considerations not previously thought of by all analysts. Its primary disadvantage is, of course, lies in the adverse social pressures present during group interaction. Group Reassessment and Group Feedback and Individual Reassessment are similar with respect to feedback and commitment. The latter is an improvement over the former in that adverse social pressures are absent. It is, however, a more complex procedure.

12.8 Some Useful Distributions for Subjective Methods

The easiest quantities for any analyst to estimate include: (a) measures of central tendencies such as median, mode, and mean, and (b) measures of dispersion such as minimum, maximum, and range. The following probability distributions lend themselves well when only this sort of information is available:

1. Uniform Distribution (Discrete or Continuous): The uniform distribution requires a knowledge only of the minimum and the maximum of the variable. Use of the uniform distribution implies, however, that the analyst has no knowledge of the relative likelihood of occurrence of the values of the random variable between these two extremes. It is, therefore, a distribution of extremely low information content.

2. Triangular Distribution: The triangular distribution requires, in addition to the minimum and the maximum, an estimate of the mode or most likely value of the variable. In spite of its simplicity, there is a great deal of information contained in this kind of a distribution. Under many conditions, it very satisfactorily approximates more complex distributions such as the normal, lognormal, and beta.

3. Trapezoidal Distribution: The trapezoidal distribution is a useful substitute for the triangular when there is a great deal of uncertainty regarding the most likely value.

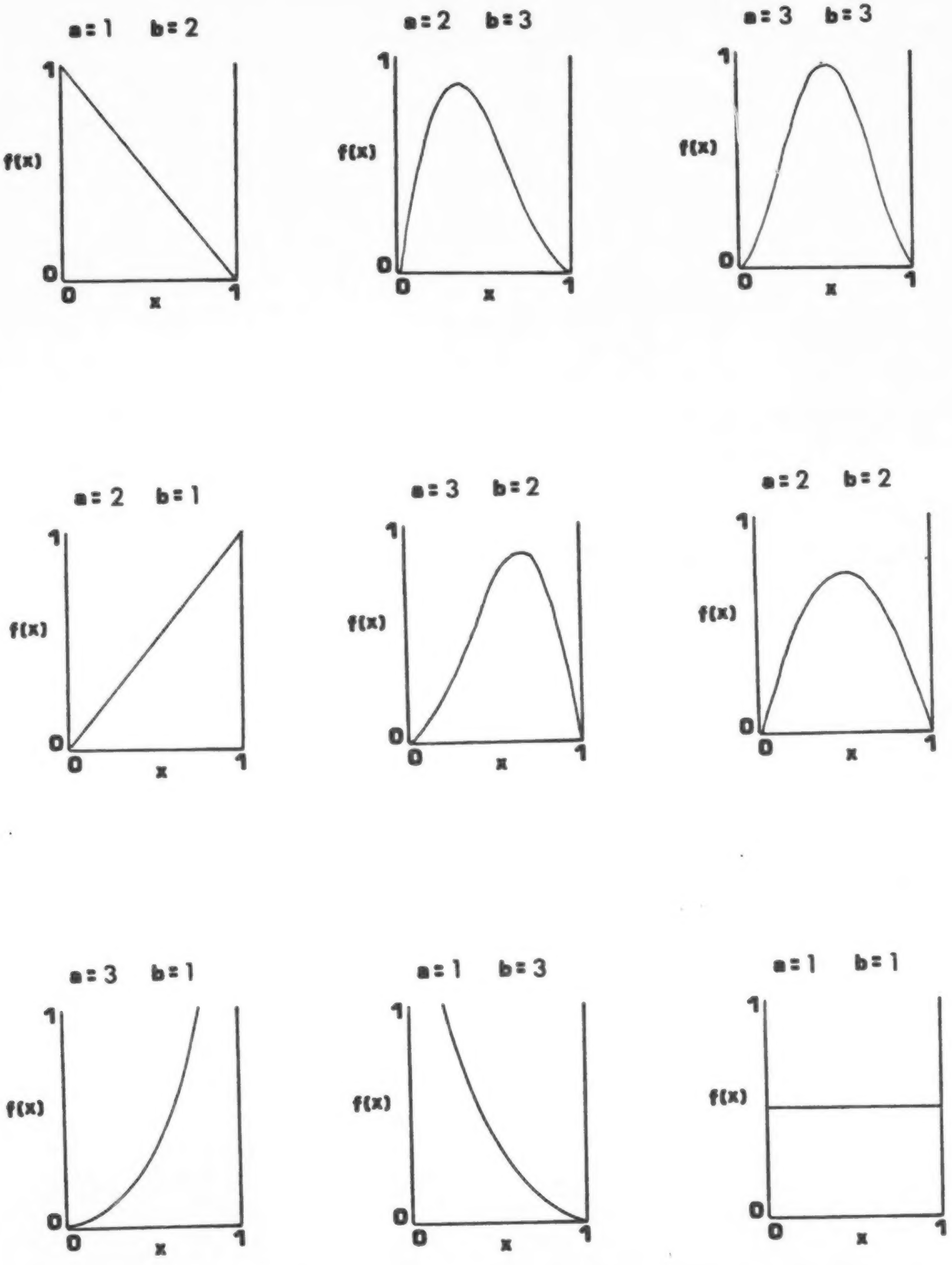


FIGURE 12.1:

SHAPES OF SOME BETA DISTRIBUTIONS

4. Beta Distribution: The beta distribution is one of the most frequently employed classical distributions for fitting subjective probability distributions (Winkler, 1972), perhaps because it fits an impressive number of different shapes (see Figure 12.1) The standard beta distribution can be fit easily using the Bayesian technique illustrated in Section 12.6. The beta distribution need not be limited to values of the random variable lying between 0 and 1, however. Any bounded interval can be rescaled to this interval. This is why MOUSE has been provided with the Empirical Beta Distribution for which the user need supply only three estimates: the most likely, and the lowest and highest value the variable can reasonably assume. These are identical to the inputs required for fitting the triangular distribution.

Another distribution of great value for subjective estimates is the step rectangular distribution. As Pouliquen (1970) states: "The distribution is an attractive one for several reasons. In the first place it takes explicit advantage of the fact that the quantification of subjective probability judgments, in both theory and practice, is based upon preference ranking. It also has the advantage that it can be drawn up by the appraiser himself. He has the freedom to choose whatever intervals he wants and to divide them into as many sub-intervals as he wants. This complete freedom of initiative, which he lacked in the case of the portrait approach (Author's note: a variation on the method of drawing the probability distribution directly), seems to help him considerably in the expression of his judgment."

REFERENCES

Bierman, H., et al., Quantitative Analysis for Business Decisions, Richard D. Irwin, Homewood, Illinois, Revised Edition, p. 1-456 (1965).

Dalkey, N.C., "Toward a Theory of Group Estimation", in H.A. Linstone and M. Turoff, Eds., The Delphi Method: Techniques and Applications, Addison-Wesley Publishing Co., p. 236-262 (1975).

Foutz, R.V., "A Test For Goodness-Of-Fit Based Upon An Empirical Probability Measure", The Annual of Statistics, Vol. 8, No. 5, p. 989-1001 (1980).

Gibbons, J.D., Nonparametric Methods for Quantitative Analysis, Holt, Rinehart and Winston, New York, p. 1-463 (1976).

Huber, G.P., "Methods for Quantifying Subjective Probabilities and Multi-Attribute Utilities", *Decision Sciences*, Vol. 5, p. 430-458 (1974).

Pouliquen, L.Y., *Risk Analysis in Project Appraisal*, World Bank Staff Occasional Papers Number Eleven, The John Hopkins University Press, Baltimore, p. 1-79 (1970).

Ruzicka, R.K., "Weibull Distribution Analysis - Short Method", *Journal of the Electronics Division, Amer. Soc. for Quality Control*, Vol. 1, No. 1, p. 38-54 (1962).

Stael von Holstein, C.-A. S., *A Manual for Encoding Probability Distributions*, prepared for the Defense Advanced Research Projects Agency by Stanford Research Institute International (SRI Project 7078), p. 1-201 (1979).

Winkler, R.L., "The Assessment of Prior Distributions in Bayesian Analysis", *J. Amer. Statistical Assoc.* 62(319), p. 776-800 (1967).

APPENDIX A

RUNNING MOUSE, IMP, AND TRAP PROGRAMS

A.1 Introduction

A number of terms will be used in this Appendix that may be new to readers with little or no experience with high-level computer languages. MOUSE was written in FORTRAN because it is the most nearly universal of all scientific programming languages. There are, in the opinion of the author, much better languages but if programs are to be widely used they must be portable, i.e., it must be possible to move them from one computing environment to another with a minimum of alterations. In this regard, FORTRAN is highly standardized; in comparison, BASIC is just beginning to become standardized, and the problem with PASCAL is that it has at least three major "standards". MOUSE has been written in a subset of FORTRAN that is handled in the same way by essentially all known computer systems.

Source files are the English-like representation of a program, i.e., the text you type in on the keyboard of a terminal through an editor or on the keyboard of a keypunch machine. If this text conforms with certain syntactical rules (the rules of MOUSE, in this case), the computer's FORTRAN compiler will turn this text into code that the computer is able to understand. Computers cannot understand source code; it must first be compiled into a form (i.e., "machine language") that can be understood by the machine being used. Machine code, which generally is not human-readable, resides in a so-called object file. In other words, the source file is what you write, and the object file is the destination of the translated (for the computer) source code.

Although every program is different, there are still commonalities. Most programs, for example, contain input and/or output operations, simple arithmetic functions such as addition or subtraction, and more complicated routines such as logarithms and square roots. In order to avoid having to compile these common elements over and over again, every FORTRAN system has an internal library which contains compiled versions of these operations. Before a compiled program can be run, the FORTRAN system must examine the program, determine which of these standardized operations are called for, and attach the required modules from the system library to the compiled main program. This process is called "linking", and the final product is a file which is "executable", i.e., it is in machine-readable code with all the required operations, functions, and subroutines appended. Note that not all of the system's library is linked with the user's main program; only those modules that are actually required are linked. Note that since MOUSE is a collection of functions and subroutines, in its compiled form it can be thought of an external library of standardized modules available during the linking process.

Error messages can be sent by the computer during any of the processes described above, i.e., during compiling, linking, and/or running. The last-named are referred to as "run-time" messages. Messages issued by the compiler describe the detected error and the location of the error in the program. If, at the time of compilation, a listing file is requested, the listing file will write out these error messages immediately after the line in your program in which the error occurred. This is, of course, most useful in debugging programs. Run-time messages are sent to your terminal or will appear on your output listings if batch mode is used. These are usually more difficult to interpret; although the type of error is reported by the computer, its exact location is generally not clear. However, as described in Chapter 11, Section 11.3, MOUSE itself provides run-time error messages that most likely will prove more helpful than those provided by the computer's FORTRAN system. A summary of MOUSE run-time error messages can be found in Appendix C.

A.2 Running CHECKER, IMP, TRACER, and TRAP On PC-DOS or MS-DOS Computers

To run these programs, simply type "CHECKER", "IMP", "TRACER", or "TRAP". TRAP takes input only from the keyboard, CHECKER and TRACER require an input file, the name of which you will be required to supply, and IMP takes input either from the keyboard or a data file called "IMP.DAT". The default output files for CHECKER, IMP, TRACER, and TRAP are "CHECKER.OUT", "IMP.OUT", "TRACETST.FOR", and "PROGRAM.OUT", respectively. If these files already exist, you will be asked whether you wish them to be overwritten or not. If not, you can supply any name you choose.

A.3 Running MOUSE On PC-DOS or MS-DOS Computers

Since MOUSE consists of various functions and subroutines, not all of which are needed for every program, these functions and subroutines are placed into a MOUSE library file (MOUSE.LIB) where they can be linked with a compiled source file. (See Appendix B, Section B.2.) The procedure is as follows:

Step #1. Using a text editor, place your program into a source file with extension FOR, e.g., PROGRAM.FOR.

Step #2. Using your FORTRAN compiler, compile this source program, producing an object file with extension OBJ, e.g., PROGRAM.OBJ

Step #3. Using your FORTRAN linker, link the object file with the MOUSE library file, producing an executable file with the extension EXE, e.g., PROGRAM.EXE.

Step #4. Run the executable file by typing the name of the executable file, e.g., "PROGRAM".

The actual operation will vary from compiler to compiler, but the following is a typical example using the Microsoft FORTRAN system, assuming that we have a MOUSE source file called PROGRAM.FOR:

```
Compilation:    FL /c /Fs /FPi PROGRAM.FOR
```

where FL is the Microsoft compilation command, /c is a switch that instructs the compiler to compile <u>without</u> linking, /Fs will produce the listing file PROGRAM.LST, and /FPi is a switch that instructs the compiler to compile generating in-line instructions while selecting the emulator math package (meaning that the program will run whether or not the computer has a math coprocessor, i.e., if the computer does have a math coprocessor, it will be used.)

```
Linking:        LINK PROGRAM,,,MOUSE.LIB

Running:        PROGRAM
```

Most MOUSE problems using the Microsoft FORTRAN system are compiled, linked, and run in his manner, although for certain very large problems other switches may have to be used (such as the /Gt[number] compiler switch). In the above example, the compiling step will produce files PROGRAM.OBJ AND PROGRAM.LST, and the linking step will produce will produce file PROGRAM.EXE.

The Microsoft FORTRAN system has an option whereby a source program can be checked for errors <u>before</u> it is compiled, e.g.,

```
Testing:        FL /c /Zs PROGRAM.FOR
```

where the /Zs option is the switch that selects program syntax without compilation. I have a batch file (called TEST.BAT) that permits me to test a source file quickly, regardless of its name. It is a one-line file consisting of the following:

```
FL /c /Zs %1.for
```

To use this batch program on the file PROGRAM.FOR, for example, I enter:

```
TEST PROGRAM
```

A typical compiling step (the linking and running steps are the same) for the IBM (Ryan-McFarland) FORTRAN system are:

```
Compilation:    PROFORT PROGRAM.FOR /B /L >PROGRAM.LST
```

The /B option will cause all adjustable arrays to be assumed to be larger than 64K bytes, and the /L > PROGRAM.LST option will generate a standard program listing in the file called PROGRAM.LST. (Note: If you use the IBM (Ryan-McFarland) FORTRAN system, the /B option <u>must</u> be used.)

APPENDIX B

INSTALLATION AND CUSTOMIZATION NOTES FOR MOUSE

B.1 Installation

MOUSE is designed to be used with a FORTRAN-77 compiler, i.e., ANSI Standard FORTRAN(X3.9-1978), containing the following extension: Hexadecimal constants must be allowed in a DATA statement to initialize character values. Both the Microsoft and IBM (Ryan-McFarland) FORTRAN allow such constants.

The distribution disk contains the source file for MOUSE called "MOUSE.FOR". MOUSE.FOR contains a number of functions and subroutines written in FORTRAN-77 and adheres closely to that standard except for the manner of utilizing hexadecimal constants in DATA statements as mentioned above. The distribution version of MOUSE contains the form, #h, where h is a string of hexadecimal characters 0 through 9 or A through F. For example, a carriage return is represented as #0D (this is the form used by the Microsoft FORTRAN). Some compilers use a Z before a quote, e.g., Z'0D' (this is the form used by the IBM or Ryan-McFarland FORTRAN).

The following are the necessary <u>Precompilation Steps</u>. However, if anything has to be modified, it most likely will be the hexadecimal constants noted in Step #1.

Step #1: Examine the following DATA statement which can be found at the beginning of the subroutine FFREAD module in MOUSE.FOR:

```
DATA CR,LF/#0D,#0A/
```

If necessary, change the form of the hexadecimal constants to conform to the requirements of your FORTRAN compiler as discussed previously. This distribution form is that used by the Microsoft FORTRAN system.

Step #2: Examine the SUBROUTINE DATA routine in MOUSE.FOR and find the following lines:

```
ITERM=5
JTERM=6
```

This assigns the terminal <u>input</u> and terminal <u>output</u>, respectively, to logical unit number 5. If it is different from your system (both the Microsoft and IBM or Ryan-McFarland FORTRAN use these assignments), change the numbers accordingly. It is not necessary to change the logical unit number assignments found in other lines.

Step #3: The seed for the random number generator in MOUSE is found in the module SUBROUTINE DATA in the line:

```
IRAN=23931
```

This can be changed to suit the user's preferences (any large positive, odd integer generally will do).

Step #3: The value of OK6 (see Chapter 11, Section 11.2) is set in the SUBROUTINE DATA subroutine in MOUSE.FOR by:

```
OK6=.0000000001
```

This can be changed to suit the computer being used. However, the default number works well on 8088 and 80286 machines.

B.2 Compilation and Linking With Libraries

As explained in Appendix A, during linking an executable file is formed with the extension EXE. After linking, the object file is no longer needed and may be deleted. Since it does not make sense to link all of MOUSE to your program when only a small number of modules may be called for by that program, it is advisable to form what is known as a "library". This is a file containing the MOUSE modules in object form, and has the extension LIB.

There are two ways to create a MOUSE library. The easiest is to compile the file MOUSE.FOR in one step. A typical Microsoft FORTRAN system compilation step is as follows:

```
Compilation:    FL /c /Zl MOUSE.FOR
```

where the /Zl option suppresses the use of library names in object files (a useful option since MOUSE library routines are not required to contain this information). The next step is to place this module into a library called MOUSE.LIB. A typical Microsoft FORTRAN system library inclusion step is as follows:

```
Library:        LIB MOUSE +MOUSE;
```

This step will produce the library file, MOUSE.LIB. The object file formed by the compilation step, MOUSE.OBJ, is of no further use now and may be deleted. A typical compilation step using the IBM (Ryan-McFarland) FORTRAN system (the library step is the same) is as follows:

```
Compilation:    PROFORT MOUSE.FOR /B
```

In the second way to produce a MOUSE library file, you must first individually compile each of the MOUSE functions and subroutines that are present in the file MOUSE.FOR. Each of these modules must be removed from MOUSE.FOR and placed in its own file with extension FOR. The normal distribution module, GAU, for example, must be placed into a file called GAU.FOR. Again, the form of the compilation command will differ with the FORTRAN system you are using. A typical Microsoft FORTRAN system compilation step is as follows:

```
Compilation:    FL /c /Zl GAU.FOR
```

where the /Zl option suppresses the use of library names in object files (a useful option since MOUSE library routines are not required to contain this information). The next step is to place this module into a library called MOUSE.LIB. A typical Microsoft FORTRAN system library inclusion step is as follows:

```
Library:        LIB MOUSE +GAU;
```

The object file formed by the compilation step, GAU.OBJ, is of no further use now and may be deleted. When you are finished compiling all the MOUSE modules and including them into the MOUSE library, you will have a complete MOUSE library file, MOUSE.LIB. A typical compilation step using the IBM (Ryan-McFarland) FORTRAN system (the library step is the same) is as follows:

```
Compilation:    PROFORT GAU.FOR /B
```

The second method has the advantage that individual modules can be removed, modified, and re-added to the library file. [Note: Since constructing a library via the second method involves quite a bit of work (although it can be automated to some extent using batch files), the distribution version of MOUSE contains a ready-to-use, individual-module library file, MOUSE.LIB, in two versions: MOUSEMS.LIB and MOUSEIBM.LIB (for Microsoft and IBM or Ryan-McFarland FORTRAN, respectively). Eliminate the unneeded file and rename the remaining one MOUSE.LIB.]

In order to prevent accidental erasure of the MOUSE.FOR and MOUSE.LIB files, it is advisable to set the RO ("read-Only") attribute using the DOS ATTRIB program as follows:

```
ATTRIB +R MOUSE.FOR
ATTRIB +R MOUSE.LIB
```

B.3 Adding Modules to MOUSE

Since MOUSE is a modular-oriented system, i.e., it is composed of separate modules or subprograms, it is easy to add user-specific modules such as probability or interest functions or to modify modules provided that the library was formed using the individual-module method. This section describes how such modules can be added. For this section only, it is assumed that the reader has a knowledge of FORTRAN.

Suppose we wish to add a two-parameter Weibull probability distribution generator to MOUSE. Following Naylor, et al. (1966), Weibull variates can be generated by the following expression:

$$X = b(-\log r)^{1/c} \qquad [A.1]$$

where b is a scale parameter and c is a shape parameter. The standard call for a random number in MOUSE is made by RAN(IRAN) and the argument is always found in COMMON location Q8. Thus, the following function would be added at any convenient location in MOUSE.FOR:

```
C
C GENERATION OF RANDOM VARIATES FROM A WEIBULL DISTRIBUTION
C
      FUNCTION WEI(B,C)
      COMMON/Q8/IRAN
      R=RAN(IRAN)
      WEI=B*(-ALOG(R))**1./C
      RETURN
      END
```

Thus a call to WEI(2.,6.) would return a random value from the Weibull distribution with scale parameter equal to 2 and shape parameter equal to 6.

Let us now consider a financial function. Doane, et al. (1976) discuss the calculation of "annualized cost", in which one of the factors used is the depreciation factor, DPF. Defined for straight line depreciation, this factor is given as:

$$DPF = [1 - (1 + i)^{-n}]/ni \qquad [A.2]$$

where n is the depreciation lifetime and i is the interest rate.

In order to allow the use of nominal interest rates, MOUSE provides for a conversion of nominal to true interest rate by the following formula (which is the coded FORTRAN version of equation 5.10):

XI=(1.+XI/COMP)**COMP-1. [A.3]

where XI is the interest rate and COMP is the number of compounding periods. If the nominal interest rate is entered as XI in the right-hand side of equation A.3, then the value of XI returned in the left-hand side will be the true interest rate. Any interest formula, therefore, that may be compounded should first test for the value of COMP; if greater than zero, then equation A.3 should be used. Thus, the function desired is (where XN is the depreciation lifetime):

```
C
C DEPRECIATION FACTOR, STRAIGHT LINE
C
      FUNCTION DPF(XI,XN,COMP)
      IF(COMP .GT. .001) GOTO 10
20    DPF=(1.-1./(1.+XI)**XN)/(XN*XI)
      RETURN
10    XI=(1.+XI/COMP)**COMP-1.
      GOTO 20
      END
```

REFERENCES

Doane, J.W., et al., The Cost of Energy from Utility-Owned Solar Electric Systems: A Required Revenue Methodology from ERDA/EPRI Evaluations, JPL Report No. 5040-29 (ERDA/JPL-1012-76/3), Pasadena, California, p. 1-82 (1976).

Naylor, T.H., et al., Computer Simulation Techniques, John Wiley & Sons, New York, p. 86 (1966).

APPENDIX C

SUMMARY TABLES

TABLE C.1

MOUSE CONTROL LINES

Standard Mouse Control Lines

```
#1           COMMON/Q1/C80/Q2/JUMP/Q3/LOOP/Q4/SENS/Q5/IDATA/Q6/IOUT/Q7/IPASS/Q
#2          &/IRAN/Q9/IREAD/Q10/ISCR1/Q11/ISCR2/Q12/ISNAP/Q13/ITER/Q14/ITOG/Q1
#3          &/IWRIT/Q16/IXXX/Q17/OK1/Q18/OK2/Q19/OK3/Q20/OK4/Q21/OK5/Q22/OK6
#4     5     DO 10 IXXX=1,ITER
#5     10    CONTINUE
#6           GOTO(5,15), JUMP
#7     15    CONTINUE
#8           END
```

Pass Control Line

```
             DO 15 IPASS=I,K        [K is an integer]
```

Vector Loop Control Lines

```
#1           DO  100  LOOP=1,L     [L is  an integer or an integer variable]
#2     100   CONTINUE
#3           LOOP=1
```

TABLE C.2

ALPHABETICAL LIST OF MOUSE INTEREST FUNCTIONS

Function	Type	Page
Compound Interest:		
CAPREC(i,n,period)	capital recovery	73
SCA(i,n,period)	single payment compound amount	71
SPW(i,n,period)	single payment present worth	72
UPW(i,n,period)	uniform series present worth	72
UPWP(i,n,j,period)	same but with constant percentage change	72
UCA(i,n,period)	uniform series compound amount	73
SINK(i,n,period)	sinking fund	73
Gradient Series:		
GCA(i,n,period)	gradient series equivalent uniform annual amount	74
GPW(i,n,period)	gradient series present worth	74
Continuous Compounding:		
CSPW(i,n,period)	single payment present worth, continuous	76
CUPW(i,n,period)	uniform series present worth, continuous	76
Miscellaneous:		
PWSUM(vector,i,upper,inv)	present worth sum	123

TABLE C.3

ALPHABETICAL LIST OF MOUSE PROBABILITY FUNCTIONS

Function	Type	Page
AGAUL(mode,median)	lognormal, M-form	46
BETA(a,b)	beta, standard form	46
BGAU1(m1,s1,Z)	bivariate normal (1)	110
BGAU2(m2,s2,rho,Z)	bivariate normal (2)	110
BGAUL1(m1,s1,X,Y,Z)	bivariate lognormal (1)	112
BGAUL2(m2,s2,rho,X,Y,Z)	bivariate lognormal (2)	112
BIN(n,p)	binomial	49
CUNI(lower,upper)	uniform, continuous	45
DTRA(a,b,c,d)	discrete trapezoidal	51
DTRI(a,b,c)	discrete triangular	50
DUNI(lower,upper)	uniform, discrete	50
EBETA(l,m,u)	beta, empirical form	46
ECTS(name,n)	empirical, continuous	40
EDIS(name,n)	empirical, discrete	40
ERL(a,k)	erlang	49
EXPON(mean)	exponential	45
GAM(a,b)	gamma	47
GAU(mean,std)	normal	45
GAUL(mean,std)	lognormal	46
HYP(N,n,p)	hypergeometric	50
PAS(k,p)	pascal/geometric	50
POI(mean)	poisson	49
STEP(name,n)	step rectangular	43
TRA(a,b,c,d)	trapezoidal	48
TRI(a,b,c)	triangular	47

IF statements for truncated distributions: 52

```
   IF(variable .LT. lower) GOTO n
   IF(variable .GT. upper) GOTO n
   IF(variable .LT. lower .OR. variable .GT. upper) GOTO n
```

IF statement construction for composite distributions: 53

```
   variable=function1
   IF(RAN(IRAN) .LE. w) variable=function2
```

CALL AUTOE(mean,std,lambda,value)
  Exponential Autocorrelation Function 118

CALL AUTOL(mean,std,k,value)
  Linear Autocorrelation Function 117

TABLE C.4

MOUSE LINE TYPES

Line	Type	Page
1. Dimension Block		
DIMENSION name(n,m)	matrix dimension	90
DIMENSION name1(n),etc.	vector dimension	91
DIMENSION name1(n1,2),etc.	empirical variable dimen.	41
DIMENSION T1(100,3),etc.	plot dimension	98
DIMENSION T1(n1),etc.	table dimension	80
DIMENSION T1(7,n1),etc.	vector table dimension	91
2. Read Block		
CALL READ(namei,ni)	empirical variable read	41
CALL MREAD(namei,n,m)	matrix read	90
3. Specification Block		
C80=80	column output	89
SENS=c	sensitivity level	104
IOUT=IWRIT	output destination	88
ISNAP=s	snap	86
ITER=n	iteration	85
IRAN=seed	seed reset	86
4. Program Block		
CALL DTABLE(switch)	dummy table output	82
CALL DTRACE('(lname)',variable)	decimal trace	147
CALL ITRACE('(lname)',variable)	integer trace	147
CALL PLOT(xvar,'(lxname)',yvar,'(lyname)',ti)	plot output	97
CALL SEN(variable,'(lname)',m%,std%)	sensitivity control	102
CALL TABLE(variable,'(lname)',d1,d2,ni,ti)	table output*	80
CALL VTABLE(variable,'(lname)',upper,Ti)	vector table output	126
IF(statement is true) then do this	IF statement	52
GOTO n	GOTO statement	52
5. Miscellaneous		
CALL ACCUM(var1,var2,'(fname)','(info)')	accumulation line	63
CALL LIST('(lname)')	program listing	95
CALL NAME('(lname)')	name	93

* Note: In User-dimensioned tables a minus sign is required in front of ni, d1 is the lower bound of the histogram to be produced by MOUSE, and d2 (a decimal number greater than zero) is the width of the intervals desired.

APPENDIX D

MOUSE ERROR MESSAGES

10 FUNCTION PARAMETERS IN WRONG ORDER: Pertains to probability functions. Example: In the continuous uniform distribution, the call is CUNI(lower,upper). This error message would be returned if the user coded CUNI(20.,10.) instead of CUNI(10.,20.).

11 DECIMAL ARGUMENT LESS THAN OR EQUAL TO ZERO: This message is most often returned when the user has omitted the decimal point in a MOUSE function or subroutine call, e.g., if the user had coded POI(5) instead of POI(5.). (See note at end of this Appendix.) Some arguments, of course, must be positive, e.g., the standard deviation in calls such as GAU(mean,std). A code of GAU(50.,-10.) would also result in this error message.

12 ITERATIONS LESS THAN 100: Plots require a minimum of 100 iterations.

13 CORRELATION COEFFICIENT RANGE EXCEEDED: The permissible range for the bivariate normal or bivariate lognormal probability distribution has been exceeded. See Chapter 9, Sections 9.3 and 9.4.

14 DECIMAL ARGUMENT LESS THAN ONE: Some probability function arguments are required to be at least 1, e.g., the N and n in HYP(N,n,p).

15 INTEGER ARGUMENT GREATER THAN 1000: This message is most often returned when the user has used a decimal point or a floating point variable when an integer or integer variable is required, e.g., PWSUM(ANBEN,.08,XLIFE,XININV) instead of PWSUM(ANBEN,.08,LIFE,XININV). (See note at end of this Appendix.)

16 BOTH CHANGE ARGUMENTS EQUAL TO ZERO: Pertains to the Sensitivity Control line. This message is most often returned when the decimal point has been omitted from the m% and std% arguments.

17 ILLEGAL USE OF PLOT OUTPUT LINE WITHIN LOOP: A Plot Output Line was used within a Vector Loop. This is not permitted.

18 ILLEGAL USE OF TABLE OUTPUT LINE WITHIN LOOP: A Table Output Line was used within a Vector Loop. This is not permitted.

19 SNAP INTERVAL EXCEEDS ITERATIONS: The snap interval must be less than the number of iterations.

20 MISSING DATA OR NAME LINE OR DATA MISENTERED: Either a line is missing in an Empirical Variable, Correlation or Matrix data set, or else data has been entered in an incorrect field. This error can also occur if there are blank lines at the beginning of the MOUSE.DAT file.

21 BOTH LIMITS ARE EQUAL TO EACH OTHER: Pertains to probability functions and generally means that the decimal points have been omitted.

22 MODE EQUAL TO OR LARGER THAN THE MEDIAN: In a lognormal distribution, the median must be greater than the mode.

23 INTEREST RATE OR INTEREST PERIOD LESS THAN ZERO: Pertains to most interest functions and is an obvious error since neither of these quantities can be negative.

24 INTEREST PERIOD LESS THAN OR EQUAL ZERO: Pertains to the Internal-Rate-of-Return function and generally means that a decimal point or a floating point variable has been used for the "upper" argument instead of an integer or an integer variable.

25 HISTOGRAM INTERVAL EQUAL TO ZERO: Pertains to user-dimensioned tables and generally means that the decimal point has been omitted.

26 DECREASE EXCEEDS 100%: In the UPWP function, the amount cannot be decreased by more than 100%, nor can the standard deviation be decreased by more than 100% in the Sensitivity Control line.

27 ITER LESS THAN 2: A MOUSE run requires at least 2 iterations.

Note: Because of the vastly different manner in which integer and decimal data are stored in a computer, decimal variables without decimals are generally interpreted as either extremely small positive numbers or as negative numbers when passed in functions or subroutines. Integer variables, on the other hand, when mistakenly used with decimal points, are generally interpreted as numbers larger than 1000. For decimal numbers, MOUSE tests to see whether the number passed is less than OK6 (which, in the distribution version of MOUSE~~, is~~ 0.0000000001). If it is less than this number, an error is declared.

APPENDIX E

ANSWERS TO EXERCISES

Chapter 3:

1. Invalid: Variable names must start with a letter of the alphabet.
2. Invalid: Two arithmetic operators cannot appear next to each other.
3. Valid.
4. Invalid: Only a single variable name can appear on the left-hand side of an equation.
5. Invalid: Q1 is a restricted variable name.
6. Valid.
7. Valid.
8. Invalid: The right-most parenthesis is missing.
9. Valid.

Chapter 4:

1. Valid.
2. Invalid: If 100 is the most likely value, 200 cannot be the lowest value.
3. Invalid: Decimal points are not used in the n argument.
4. Valid.
5. Invalid: The probability is the first argument in the binomial probability function.
6. Valid.
7. Invalid: The lognormal probability distribution cannot have a negative mean.
8. Valid.
9. Valid.

10. Invalid: The statement should be CALL READ(D7,30).
11. There are no errors in this data line set.

12. The probability of the first value in a discrete empirical probability distribution must be greater than zero; otherwise, why put it in?

13. The first cumulative probability in a continuous empirical probability distribution must be zero.

14.

```
      200     X=TRI(10.,20.,70.)
              IF(X .LT. 15. .OR. X .GT. 60.) GOTO 200
```

15.

```
              PROD=GAU(100.,10.)
              R=RAN(IRAN)
              IF(R .LE. .3333) PROD=GAU(75.,5.)
              IF(R .GE. .6667) PROD=GAU(55.,5.)
```

16. Data Set #1: The closest fits are the exponential, lognormal, and gamma. The data were actually taken from a lognormal distribution. This exercise shows the dangers in relying solely upon goodness-of-fit tests when small data sets are involved.

    Data Set #2: The best fit is the Poisson distribution. (The data were actually taken from a Poisson distribution.)

17. The 25-point integration provides a substantially better fit than does the 13-point integration.

Chapter 5:

1. Invalid: The period argument lacks a decimal point.

2. and 3. Both are valid.

4. Invalid: The interest rate and period arguments are reversed.

5. Invalid: The second argument in the UPWP function is the time period, not the fraction change.

6. Invalid: For continuous compounding, set the switch to any positive number.

7. Valid.

8. Invalid: Interest rates cannot be negative. (An exception is the i argument in the PWSUM function; see Chapter 10, Section 10.2.)

9. Invalid: The period must be an integer.

Chapter 6:

1. Valid.
2. Invalid: The number of intervals argument should be an integer.
3. Invalid: The parentheses around the literal name are omitted, and the variable name exceeds 6 characters.
4. Valid.
5. Valid.
6. Invalid: For MOUSE-dimensioned tables, the minus sign in front of the number of intervals should not be used.
7. Valid: The scaling factor lacks a decimal point.
8. Invalid: The table interval must be greater than zero.
9. Invalid: The dimension specification should be T1(40).
10. Invalid: The number of intervals for short-form tables should be set equal to zero.
11. Valid.

Chapter 7:

1. Valid.
2. Invalid: Seed lines must contain odd integers.
3. Valid.
4. Invalid: The first index of any matrix must be IPASS.
5. Invalid: Either the row or column argument is missing.
6. The index IPASS must be used instead of I.
7. There are no errors in this data set.
8. Since there are four rows in the matrix, the second row, containing the data pair (5.,4.), must start on a new line, etc.

Chapter 8:

1. Valid.
2. Valid.
3. Valid.
4. Invalid: The standard deviation cannot be decreased by more than 100%.
5. Valid.
6. Valid.
7. Invalid: The last executable statement in a MOUSE program must be a Table Output Line.

Chapter 9:

1. Valid.
2. Invalid: The dummy variable must be the same in both cases.
3. Valid.
4.

```
     rx = .01        mx = 4.6002        sx = .0998
     ry = .04        my = 5.2787        sy = .1980
     rxy = .02
               rho1 = .6199        rho2 = .6534
               rho1' = .6176       rho2' = .6512
                       rho' = .6342
```

Chapter 10:

1. Invalid: The "upper" argument must be an integer number or an integer variable.
2. Valid.
3. Invalid: The "upper" index in the first Vector Control Line must be an integer or an integer variable.
4. Invalid: The call is VTABLE, not TABLE.
5. Valid.

6. Valid.

7. The following line is placed immediately after the ANBEN(LOOP) line in the vector loop:
   IF(LOOP.EQ.LIFE) ANBEN(LOOP)=ANBEN(LOOP)+CUNI(4000.,6000.)

8.

```
      DIMENSION T1(6),T2(26)
      COMMON/Q1/C80/Q2/JUMP/Q3/LOOP/Q4/SENS/Q5/IDATA/Q6/IOUT/Q7/IPASS/Q8
     &/IRAN/Q9/IREAD/Q10/ISCR1/Q11/ISCR2/Q12/ISNAP/Q13/ITER/Q14/ITOG/Q15
     &/IWRIT/Q16/IXXX/Q17/OK1/Q18/OK2/Q19/OK3/Q20/OK4/Q21/OK5/Q22/OK6
      ITER=1000
5     DO 10 IXXX=ITER
C
C INITIALIZE REMAINING LIFE AND TIME IN TUNNEL
C
      REMLIF=GAU(10.,2.)
      TIME=0.
200   R=RAN(IRAN)
      IF(R .LE. .3333) GOTO 220
      IF(R .LE. .6667) TIME=TIME+EXP(3.)
      IF(R .GT. .6667) TIME=TIME+GAUL(6.,2.)
C
C CHECK TO SEE IF RAT DIES BEFORE ESCAPING
C
      IF(TIME .GT. REMLIF) GOTO 210
      GOTO 200
C
C THIS RAT DIED IN A TUNNEL
C
210   TIME=REMLIF
      SURV=0.
      GOTO 230
C
C THIS RAT ESCAPED
C
220   SURV=1.
230   CALL TABLE(SURV,'(SURVIVOR)',,,0,T1)
      CALL TABLE(TIME,'(TIME)',0.,1.,20,T2)
10    CONTINUE
      GOTO(5,15), JUMP
15    CONTINUE
      END
```

Note that this is a terminating simulation (see Chapter 7, Section 7.3). The simulation (for one rat) ends when either the rat dies or escapes the cage. The initial condition in this problem is that only one rat is present in the cage at the start. Additional rats at the start would affect the results if, for example, a rat is not permitted to enter a tunnel if another rat is already there.

```
      9.
      DIMENSION A1(20),A2(20),A3(20),B1(100),B2(100),B3(100)
      DIMENSION T1(7,20),T2(7,20),T3(7,20)
      COMMON/Q1/C80/Q2/JUMP/Q3/LOOP/Q4/SENS/Q5/IDATA/Q6/IOUT/Q7/IPASS/Q8
     &/IRAN/Q9/IREAD/Q10/ISCR1/Q11/ISCR2/Q12/ISNAP/Q13/ITER/Q14/ITOG/Q15
     &/IWRIT/Q16/IXXX/Q17/OK1/Q18/OK2/Q19/OK3/Q20/OK4/Q21/OK5/Q22/OK6
      ITER=100
5     DO 10 IXXX=ITER
C
C INITIALIZE NUMBER OF PEGS
C
      DO 200 J=1,100
      B1(J)=1.
      B2(J)=0.
200   B3(J)=0.
      DO 100 LOOP=1,20
C
C SELECTION FOR BOARD #1 TO BOARD #2
C
      DO 300 J=1,15
      I=DUNI(1.,100.)
      IF(B1(I) .NE. 0.) B2(I)=1.
300   IF(B1(I) .NE. 0.) B1(I)=0.
C
C SELECTION FOR BOARD #2 TO BOARD #3
C
      DO 400 J=1,5
      I=DUNI(1.,100.)
      IF(B2(I) .NE. 0.) B3(I)=1.
400   IF(B2(I) .NE. 0.) B2(I)=0.
C
C SUM PEGS ON EACH BOARD
C
      A1(LOOP)=PWSUM(B1,0.,100.)
      A2(LOOP)=PWSUM(B2,0.,100.)
      A3(LOOP)=PWSUM(B3,0.,100.)
C
C ACCUMULATE NUMBER OF PEGS/BOARD PER HOUR IN A VECTOR
C
      CALL VTABLE(A1,'(BOARD1)',20,T1)
      CALL VTABLE(A2,'(BOARD2)',20,T2)
      CALL VTABLE(A3,'(BOARD3)',20,T3)
100   CONTINUE
      LOOP=1
      CALL DTABLE
10    CONTINUE
      END
```

This is another terminating simulation since the operation definitely is over when all of the pegs are moved to Board #3. The initial conditions are the number of pegs on each board at the start of the simulation. Note that it is not necessary to use MOUSE Control lines 6 and 7 in this problem.

Chapter 11

1. Valid
2. Invalid: Trace lines for variables that appear within a vector loop must be placed within the loop itself.
3. Valid.

APPENDIX F

MOUSE RANDOM NUMBER TESTER

F.1 Introduction

The results of any simulation based upon "random" numbers are reliable only to the extent that the numbers are truly random. Unfortunately, many of the random number generators in use today are not very satisfactory, either because of intrinsic weaknesses or because of the selection of an inappropriate seed number. RANDOM is a MOUSE utility that subjects any set of random numbers (up to 5000 numbers) to a series of empirical and theoretical tests for randomness.

Input to the program consists of a file, called RANDOM.DAT, containing the random numbers (i.e. numbers continuously distributed uniformly, 0 < X < 1) in any arbitrary format. It is suggested that the set contain between 1000 and 5000 numbers. The following is a MOUSE program that will generate 5000 random numbers for testing:

```
C
C RANDOM NUMBER GENERATOR
C
      COMMON/Q1/C80/Q2/JUMP/Q3/LOOP/Q4/SENS/Q5/IDATA/Q6/IOUT/Q7/IPASS/Q8
     &/IRAN/Q9/IREAD/Q10/ISCR1/Q11/ISCR2/Q12/ISNAP/Q13/ITER/Q14/ITOG/Q15
     &/IWRIT/Q16/IXXX/Q17/OK1/Q18/OK2/Q19/OK3/Q20/OK4/Q21/OK5/Q22/OK6
      CALL DATA
      ITER=5000
5     DO 10 IXXX=1,ITER
      R=RAN(IRAN)
      CALL ACCUM(R,0.,'(RANDOM.DAT)','(RANDOM NUMBER SET FOR TESTING)')
      CALL DTABLE(2)
10    CONTINUE
      GOTO(5,15), JUMP
15    CONTINUE
      END
```

This program will place the title, "RANDOM NUMBER SET FOR TESTING", and 5000 random numbers in a file called RANDOM.DAT.

The program is called merely by typing "RANDOM",and starts by requesting three pieces of information:

(1) A title or comment line (this is different from the one that is in the data file, RANDOM.DAT);
(2) The name of an output file (the default is RANDOM.OUT). If the file exists, you will asked whether the file is to be overwritten or not.

RANDOM will report in the output file the first 10 numbers that were in the RANDOM.DAT file (for a quick validation that the numbers were correctly read in). An equidistribution test, which tests whether the numbers are indeed uniformly distributed over the interval (0,1), is then automatically carried out. This test involves dividing the interval (0,1) into five equal divisions and computing the chi-square statistics with 4 degrees of freedom and (in parentheses after the chi-square statistic) the probability level of the statistic.

A menu then is presented, offering the following nine optional random number tests:

(1) Gap and Two-dimensional serial test
(2) Kolmogorov-smirnov test
(3) Poker test
(4) Permutation test
(5) Runs test
(6) Maximum-of-T test
(7) D-square test
(8) Serial correlations and correlograms
(9) Lag test

F.2 Gap Test

A gap is defined as 0<a<b<1. The length of the gap is the number of numbers in the sequence (of random numbers to be tested) between occurrences of numbers that lie within the interval (a,b). The user is requested to enter the values of a and b for the test. The gap test can be repeated by the user with different values of a and b. Like the frequency test, the gap test also tests the uniformity of distribution. The Chi-square statistic and its associated probability level (in parentheses after the chi-square statistic) are reported.

F.3 Two-dimensional Serial Test

This test concerns the degree of randomness between successive pairs of numbers. It calculates a chi-square statistic with 24 degrees of freedom (and, in parentheses after the chi-square statistic, the probability level of the statistic) and requires no additional input from the user.

F.4 Kolmogorov-Smirnov Test

This is a nonparametric test and is applied in three different ways. First, a KS test is performed on the entire sequence and KN+ and KN- statistics are computed. Then the entire sequence is subdivided into ten subsequences and KS test is applied to each of the subsequences, resulting in ten KN+ and ten KS-statistics. Finally the KS test is applied to the sequence of these ten KN+ and KN- statistics. No additional input is required of the user for this test. Following each KN statistic the probability level of the statistic is reported (in parentheses). Note: This is a lengthy test.

F.5 Poker Test

The poker test the randomness of digits of random numbers. This test can be performed either on a single digit or on more than one digit at a time. The routine performs the poker test on the first digit (the most significant digit in a uniform random number), the second digit, and the first two digits combined and the first three digits combined. In order to carry out this test the entire sequence is subdivided into n-tuples. In the RANDOM program, n can be any number between 2 and 11. The program requests this number from the user and can be repeated for different values of n. Chi-square statistics (and their associated probability levels) are reported.

F.6 Permutation Test

This test is also performed on n-tuples (n =<6) and is based upon the premise that in each n-tuple the elements may have n! possible orderings. The number of times each ordering is met with is recorded and a chi-square test with n! degrees of freedom is applied. For example, for n=3, a,b and may have six possible orderings (e.g., a<b<c, b<c<a, etc.) and we count the number of times each ordering occurs in the entire sequence. The test can be repeated with different values of n. Chi-square statistics (and their associated probability levels) are reported.

F.7 Runs Test

This is one of the most often applied tests. The version supplied in RANDOM is based on the length of the monotone subsequences of the original sequence, i.e., segments that are increasing or decreasing. It is of two types: Runs Up and Runs Down. For example, in the sequence 1,2,4,7,2,1,3, Runs Up are /1,2,4,7/2/1,3/ and Runs Down are /1/2/4/7,2,1/3. The length of the segments are called the "length of the runs". Chi-square statistics (and their associated probability levels) are reported. The test requires no additional input from the user.

F.8 Maximum-of-T Test

This test is based upon the distribution of $V_j=\max(r_{tj},r_{tj+1},...,r_{tj+t-1})$ where the r's are a segment of the set of random numbers to be tested. The KS test is then applied to the sequence, V_0, V_1,, v_2,,...v_{n-1}. The maximum test of T uniform random numbers is also called the test of t-tuples and is considered to be a more stringent test than the basic equidistribution test. Kolmogorov-Smirnov statistics (and their associated probability levels) are reported. The test can be repeated with different values of T. Note: Like the Kolmogorov-Smirnov test, this is a lengthy test.

F.9 D-square Test

This test is based on the fact that consecutive random numbers may be considered as coordinates of two points and the Euclidian distance between the two points can be obtained. Chi-square statistics can be computed based on the differences between the observed and the expected frequencies of these distances. Chi-square statistics (and their associated probability levels) are reported for the test. The test requires no additional input from the user.

F.10 Serial Correlations and Correlograms

Serial tests are used to check the degree of randomness between successive numbers in a sequence. In RANDOM, the serial correlation (a number between -1 and 1), its expected value and 95% confidence interval are computed. Also computed are the autocorrelations for a specified lag, K (K=<50). These autocorrelation values may be plotted by the user against the lag length to prepare the correlogram. Different values of K can be tested by the user.

F.11 Lag Test

Somewhat related to the serial correlation (which is a measure of the amount that r_{j+1} depends upon r_j)), the lag test involves the sum of lagged products, $r_i r_{i+k}$. If there is no correlation between r_i and r_{i+k}, the test statistic is distributed normally. RANDOM reports the test statistic (and, in parentheses, its associated probability level), and its expectation and variance.

If a set of random numbers does not pass all or most of the tests available in RANDOM, the user might try changing the seed value, IRAN (see Appendix B, Section B.1). RANDOM was applied to 5000 random numbers generated by the random number generator distributed with MOUSE, using the program shown at the beginning of Appendix E and the seed value of 23971. It passed all tests except the Kolmogorov-Smirnov test for the KN+ subsequence number 10 (significance = 0.07) and for KN+ on the Maximum-of-T test (significance level = 0.02).

INDEX

Accumulation Line, 63,108,116,119,199

Adding Modules (see Modules, Adding)

AGAUL, 46

Algebraic Equations, 1

Arithmetic Operations, 35-38

Autocorrelation
    Exponential Autocorrelated Function, 118-119
    General, 116
    IMP Analysis, 118-119
    Linear Autocorrelated Function, 117-118

Average Opportunity Loss, 79

Average Value Approach, 2

Bayesian Methods, 167

Best Value Approach (see Average Value Approach)

BETA, 46

Bets (see Gambles)

BGAU1, 110

BGAU2, 110

BGAUL1, 112

BGAUL2, 112

BIN, 49

Bivariate Distributions
    Analysis, IMP, 107-109
    Analysis, Bivariate Normal & Lognormal, 112-116
    Lognormal, 112-114
    Normal, 108-112

Blocks, 31-32,35

C80, 89

CALL ACCUM, 63,108,116,119,199

CALL AUTOE, 118

CALL AUTOL, 117

CALL DTABLE, 82

CALL DTRACE, 147

CALL ITRACE, 147

CALL LIST, 95

CALL MREAD, 90

CALL NAME, 93

CALL PLOT, 97

CALL READ, 41

CALL SEN, 102

CALL TABLE, 80-82

CALL VTABLE, 126

Camp-Meidell Inequality, 81

CAPREC, 73

Case History, 141

Cash Flow, Non-Uniform, 121

CHECKER, 137-141

Chi-Square Test, 162

Classical Probability Distributions
(see Probability Functions)

Coefficient of Determination, 99

Coefficient of Variation, 23,79

Column Output Line, 88

Comment Lines, 31

Compiling, 176,181-182

Complete Enumeration (see Direct Enumeration)

Conservative Approach, 2

Continuation Lines, 38

Control Lines
  MOUSE, 31
  Pass, 92
  Summary, 185
  Vector Loop, 124

Correlated Stochastic Variables
  Bivariate Lognormal, 112-114
  Bivariate Normal, 108-112
  Chaining, 111
  Linear, 106-108

Correlation (see Correlated Stochastic Variables)

Correlograms, 202

Correlation Coefficient
  Bivariate Lognormal, 112
  Bivariate Normal, 110
  Linear, 106

Correlation Function, 106,110-114

CSPW, 76

CUNI, 45

CUPW, 76

Customization of MOUSE, 180-181

Data Files, IMP, 56-57

Data Files, MOUSE, 41-43

Data, Fitting Distributions to
  (see Probability Distributions, Fitting)

Debugging, 147

Decimal Points, Use of in MOUSE, 35-36

Delphi Method, 170

Device Number (see Logical Unit or Device Number)

DIMENSION
    Block, 31
    Empirical Variable, 41
    Matrix, 90
    Plot, 98
    Table, 80
    Vector, 91
    Vector Table, 129

Direct Enumeration, 5

D-Square Test, 202

DTRA, 51

DTRI, 50-51

Dummy Tables, 82

DUNI, 50

EBETA, 46-47

ECTS, 40-42

EDIS, 40-42

Elicitation of Subjective Probability Distributions, 163-168

Empirical Probability Functions
  (see Probability Functions)

Empirical Variable
    Data Line, 41-42
    Dimension Line, 41
    Read Line, 41

Enumeration (see Direct Enumeration)

Equations, MOUSE, 37-38

Equidistribution Test, 200

Equivalent Prior Sample Method, 166

ERL, 49

Errors, 137

Error Messages, 137,189-191

Executable File, 176-177

Exercises, Answers to, 192-198

Expected Value of Perfect Information, 79

EXPON, 45

Fitting Probability Distributions to Sample Data
  (see Probability Distributions, Fitting)

FORTRAN, 13-14,176

Foutz Statistic and Critical Values, 60-61

Frequency Distributions (see Histograms, Examples of)

Functions, Algebraic, 1

GAM, 47

Gambles, 167

Gap Test, 200

GAU, 45

GAUL, 46

GCA, 74

Goodness-of-fit Tests, 60,99,162-163

GOTO Statement, 52

GPW, 74

Group Decision Processes, 170

Group Feedback and Individual Reassessment, 170

Group Reassessment, 170

Histograms, Examples of, 26,30

HYP, 50

IF Statements, 52-53,101,125

IMP
    Autocorrelation Analysis, 63
    Bivariate Distribution Analysis, 114-116
    Fitting Distributions to Sample Data, 55-63
    Integrating Empirical Continuous Distributions, 63-66

Installation of MOUSE, 180

Interest Functions
    Capital Recovery Factor, 73
    General, 71
    Gradient Series Equivalent Uniform Annual Amount Factor, 74
    Gradient Series Present Worth Factor, 74
    Present Worth Sum, 123-124
    Single Payment Compound Amount Factor, 71
    Single Payment Present Worth Factor, 72
    Single Payment Present Worth Factor with Continuous Compounding, 76
    Sinking Fund Factor, 73
    Uniform Series Compound Amount Factor, 73
    Uniform Series Present Worth Factor, 72
    Uniform Series Present Worth Factor with Constant Percentage Change, 72
    Uniform Series Present Worth Factor with Continuous Compounding, 76

Interest Rates
    Compound, 74
    Continuous, 75-76
    Effective, 74-77
    Force of Interest, 76
    Nominal, 74-77

IOUT, 88

IPASS, 92,101

IRAN, 86

ISNAP, 86

ITER, 85

Iteration Number Line, 85

Kolmogorov-Smirnov Test, 61,162,201

Kurtosis, 129-131

Lag Test, 203

Languages, Computer
    General Purpose, 14
    Simulation, 9

Level of Significance (see Significance, Level of)

Libraries, 176-177,181-182

Line Types, MOUSE, 188

Linking, 176,181-182

Listing a Program File, 95

Listing File, 177

Literals, 83

Logical Unit or Device Number, 88,180

Loop, Vector, 121,124

Mathematical Combination, 169

Matrix
    Data Lines, 90-91
    Dimension Line, 90
    Read Line, 90
    Reading Data from Program, 90

Maximum-of-T Test, 202

Model Sampling, 7

Models, Mathematical, 1

Modifications to Distribution Programs
    Hexadecimals, 180
    Logical Unit or Device Numbers, 180
    MOUSE Random Number Seed, 181
    OK6, 143,181

Modules, Adding
    Probability, 183
    Interest, 183

Monte Carlo Simulation, 7

MOUSE.HLP, 159

MSHELP, 158

Multiple Runs, 89-93

Name Line, 93

Names
    Literal, 83
    Program, 93
    Restricted, 38
    Variable, 35-36

Non-linear Functions, 3,15

Non-Uniform Cash Flows, 121

Null Hypothesis, 61

Object File, 176

OK6, 143,181

Output Destination Line, 88

Overflow, Arithmetic, 137

Parameter, 89,101

PAS, 50

Pass,
    General, 92-93,101
    Control Line, 92

Permutation Test, 201

Plots
    Dimension Line, 98
    IF Statements in, 101
    Output Line, 97
    Scaling, 99
    Vectors in, 135

POI, 49

Point Estimates, 1

Poker Test, 201

Present Worth Sum Function, 123-124

Probability Calculus, 7

Probability Distributions, Fitting, 55-63,160

Probability Distributions, Integrating, 63-66

Probability Functions
    Beta, Empirical Form, 46-47
    Beta, Standard Form, 46,173
    Binomial, 49
    Bivariate Lognormal, 112-114
    Bivariate Normal, 108-112
    Composite, 52-54
    Empirical, Continuous, 40-42,172
    Empirical, Discrete, 40-42,172
    Erlang, 49
    Exponential, 45
    Gamma, 47
    Hypergeometric, 50
    Lognormal, 46
    Lognormal, M-form, 46
    Normal, 45
    Pascal/Geometric, 50
    Poisson, 49
    Pseudo-Continuous, 63
    Step Rectangular, 43-44,174
    Trapezoidal, 48,172
    Trapezoidal, Discrete, 51,172
    Triangular, 47,172
    Triangular, Discrete, 50-51,172
    Truncated, 51-52
    Uniform, Continuous, 45
    Uniform, Discrete, 50
    Weibull, 183

Probability Generators, Methods, 55

Program Block, 31-32,35

Program Listing Line, 95

Programs, Examples (see Waste Oil Reclamation
    Project Problem and Regulatory Problem)

Project Life, Stochastic, 124

Propagation of Error, 7

Pseudo-Continuous Probability Distribution, 63

PWSUM, 123-124

R-Square, 99

RANDOM, 199

Random Number
    Generator, 199
    Seed, 86
    Tester, 199

Random Variables
    Continuous, 40-43
    Discrete, 40-43

Read Block, 31-32,35

Read-Only Files, 182

Regression, 99

Regulatory Problem
    Basic Problem, 22-30
    Modification #1, 98

Restricted Names, 38

Rules of MOUSE, 35-38

Running Programs
    CHECKER, 177
    IMP, 177
    MOUSE, 177-179
    TRACER, 177
    TRAP, 177

Run-Time Error Messages, 143

Runs Test, 202

Salvage Value, 125

Sample Data, Fitting Distributions to, 160

SCA, 71

Scaling
    Plots, 99
    Tables, Standard, 83
    Tables, Vector, 129

Seed Reset Line, 86

Semi-Empirical Probability Function
  (see Step Rectangular Function)

SENS, 104

Sensitivity Analysis, 2,9,101-105

Sensitivity Control Line,
    General, 102
    Use with correlated variables, 111-112
Sensitivity Level Line, 104

Serial Correlations and Correlograms, 202

Significance Levels, 60,162

Simulation
    Monte Carlo, 7
    Steady State, 85-86
    Terminating, 85-86

Single Value Approach, 1

SINK, 73

Skeleton Program Lines, 157

Skewness, 129-131

Snap Interval Line, 88

Source File, 176

Specification Block, 31-32,89,95

SPW, 72

Steady State Simulation, 85-86

STEP, 43-44

Stochastic Variable
    Data Line, 41-42
    Dimension Line, 41
    Direct Method, 39,125,126
    General, 39-40
    Indirect Method, 39
    Read Line, 41

Subjective Probability Distributions
    Biases, 163,168-169
    Elicitation Methods, Improving, 168-172
    Fixed Interval Methods, 165
    Miscellaneous Methods, 166
    Portrait Approach, 174
    Useful Distributions for, 172-174
    Variable Interval Methods, 163

Tables, Dummy, 82

Tables, Mixing, 84

Tables, Standard
    Dimension Line, 80-81
    General, 79
    MOUSE-dimensioned, 79-81
    Output Line, 80-81
    Scaling, 81
    Short Form, 82
    User-dimensioned, 79,81-82
    Vector, use in, 135

Tables, Vector
    Dimension Line, 129
    Output Line, 126
    Scaling, 129

Tchebycheff Inequality, 81

Terminating Simulation, 85-86

Titles (see Name Line)

TRA, 48

Trace Lines
    Decimal, 147
    integer, 147

TRACER, 151

TRAP, Using, 151-158

TRI, 47

Troubleshooting, 137

Two-dimensional Serial Test, 201

UCA, 73

Uncertainty
    Failure to Deal With, 1
    Nature of, 9
    Solutions to, 5
    Traditional Approaches, 2-3

Uncertainty Analysis, 2-9

UPW, 72

UPWP, 72

Variables
  Decimal, 35-36
  Integer, 35-36
  Stochastic, 39

Variance, 3

Vector
  Dimension Line, 91,123
  Elements, Calculating of, 124-125
  Plots, Using Vectors in, 128,131
  Substitution for Standard Table, 131-135

Vector Loop
  IF Statements in Vector Loops, 125
  LOOP, 124
  Order of Statements in Vector Loops, 124,126
  Standard Tables and Plots, prohibited in, 135

Waste Oil Reclamation Project Problem
  Basic Problem, 18-25,33-34
  Modification #1, 87
  Modification #2, 122
  Modification #3, 134

Weighing of Several Estimators, 169-170